SUPER ROM

Lucid Spreadsheet **Write ROM** **Database** **Outliner**

"The Tandy Portable Drive with one button boot!"

NOW AVAILABLE ON THE M-200

NOW YOU CAN REALLY HAVE IT ALL!

All on one ROM. Truly the finest four programs available for the Model 100 — guaranteed. Try it for 30 days. If you aren't blown away by the excellence return it for a full refund.

$199.95

The four best programs for the Model 100 all on one ROM. 32K of power without using any RAM for program storage. This is the PCSG Snap-In ROM that just presses easily into the little ROM socket in the compartment on the back. You access the four right from the main menu like built-ins.

Write ROM — the definitive word processor for the Model 100. Function key formatting or dot commands. Search and replace. Library feature — inserts words, phrases or whole documents into text from just a code. MAP lets you see a picture of your document. In all there are 60 features and functions. No one can claim faster operation. FORM lets you create interactive forms with on-screen prompts that you can answer from the keyboard. Nothing else for the Model 100 compares with the features of Write ROM. Exactly the same as the Write ROM sold as a single program. Infoworld says it "makes the Model 100 a viable writing unit . . . surpassed our highest expectations for quality and clarity."

Lucid Spreadsheet: This is the one PICO magazine says "blows Multiplan right out of the socket" and Infoworld performance rated as "excellent" and said "makes the Model 100 compute." Gives you features you cannot get with Lotus 123. Lets you build spreadsheets in your Model 100 that would consume 140-150K on a desktop. Program generating capability with no programming knowledge required. Variable column widths. Includes find and sort with function key control. It's fast, recalculates like lightning. No feature has been taken from the original, only new ones added.

Database: This is a relational data base like no other. You can do everything from mailing lists to invoices. No complicated pseudo-coding, you create input screens as simply as typing into TEXT. You are not limited by size; you can have as large an input screen as you wish. Prints out reports or forms, getting information from as many files as you like. Complete math between fields. Total interface with Lucid worksheets.

Outliner: Does everything that Thinktank does on a PC but a whole lot better. Includes a Sort for your headlines. Lets you have headlines of up to 240 characters. Has cloning, hoisting and sideways scroll up to 250 characters. Like Lucid, this one sets a new standard for outliners. This is the way to plan and organize your projects.

Present Lucid and Write ROM owners can upgrade for **$125**. If you have both it's **$100**.

As usual PCSG sells the Super ROM on a thirty day guarantee. If for any reason you are not satisfied, simply return it for a full refund.

We are excited about this product. Super ROM gives the Model 100 the true power of a desktop. No other multi-program ROM has software that compares. But don't take our word for it. We invite you to make that comparison yourself. Priced at $199.95 on Snap-In ROM.

Got stuck with somebody else's multi-ROM? We'll upgrade it for $100.

(214) 351-0564

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd., #206, Dallas, TX 75229

MC, Visa, American Express, Check, or C.O.D.

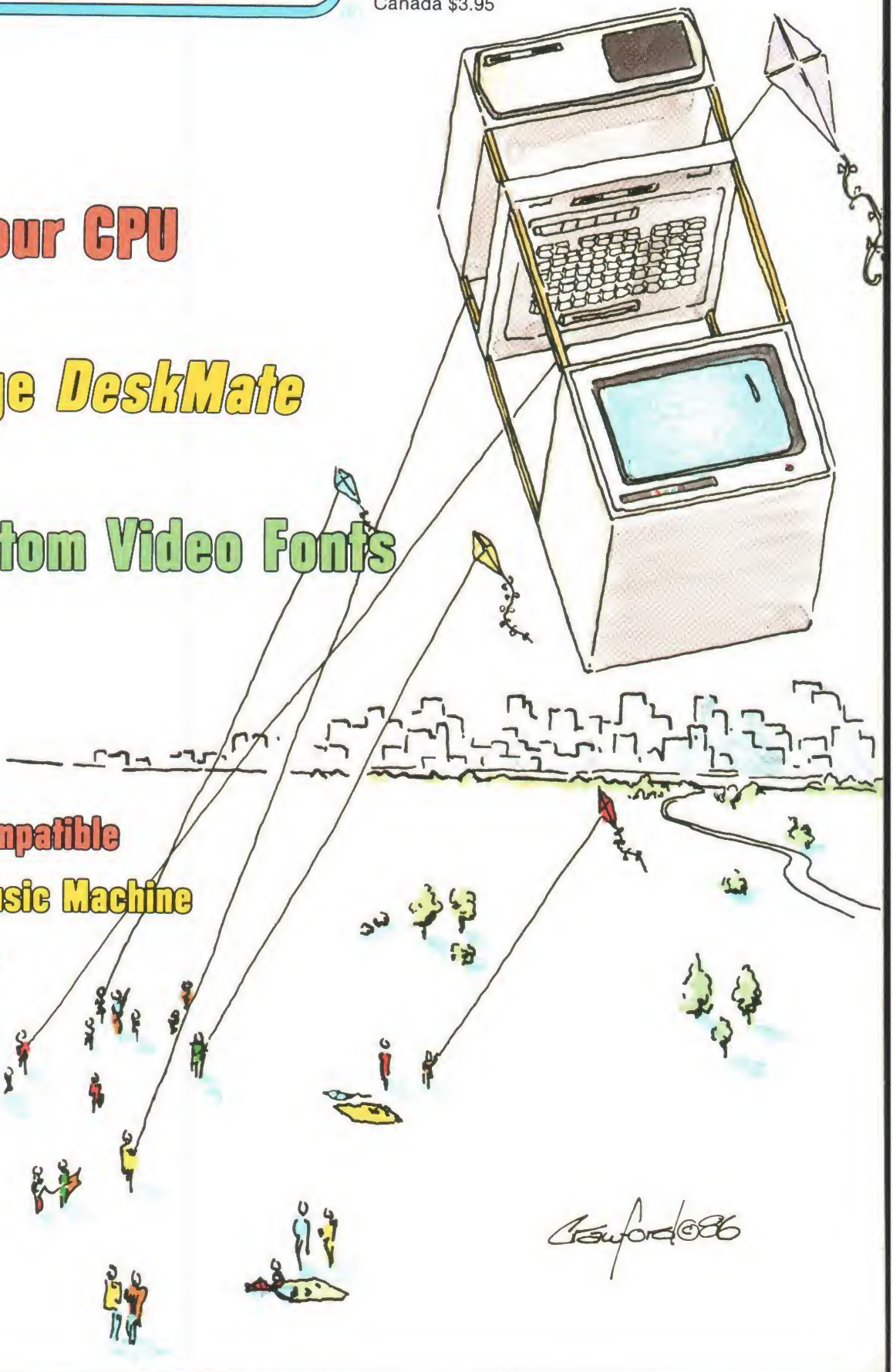
PCM

For Tandy's 100 and 600 Portables and 1000, 1200, 2000 and 3000 MS-DOS Computers

The Personal Computer Magazine for Tandy® Computer Users

Vol. III No. II
May 1986

U.S. $3
Canada $3.95



Soup Up Your CPU

Turbocharge *DeskMate*

Create Custom Video Fonts

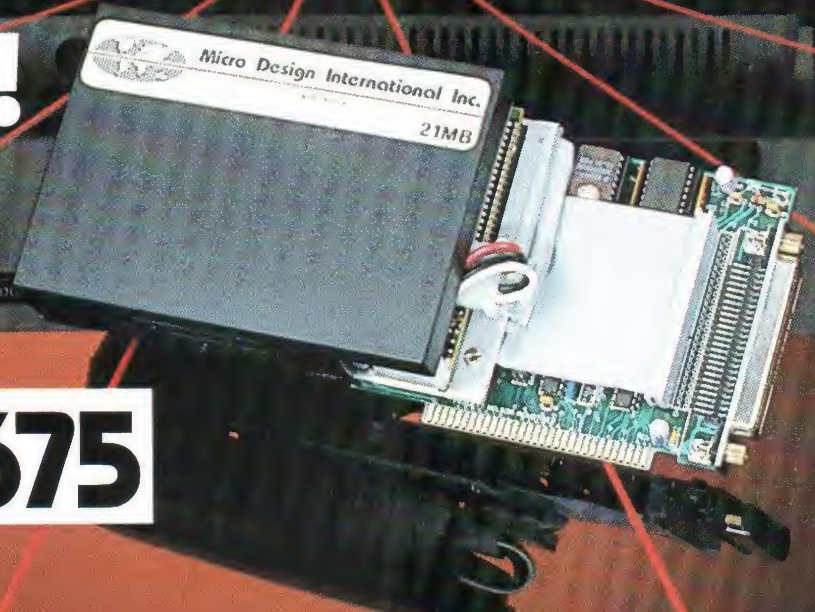
Plus

kes a Compatible

velous Music Machine

's Helper

21 MEGABYTE SCSI HARD DISK CARD



NEW!!!

for your TANDY 1000

only $675

IMAGINE!!! MDI'S **21 MEGABYTE HARD DISK CARD** simply plugs into your system and is ready to run!!! The hard disk card also contains a...

"FREE SCSI PORT":

A **SCSI** port which allows you to tie (daisy chain) up to seven external (peripheral) devices to your computer.

INCLUDED **FREE**

Direct/Assist	$49.95
R/Assist	$49.95
Light Speed	$79.95
SCSI Board	$190.00

Tandy 1000 Multifunction Board $255

NOW IN OUR 8TH YEAR

ORDER LINE **800-228-0891**
305-677-8333

MasterCard VISA

1 YEAR WARRANTY, 30 DAY MONEY BACK GUARANTEE

Micro Design International, Inc.

6566 University Boulevard, Winter Park, Florida 32792 305-677-8333 telex 332559 MDII ORLUD

PCM

The Personal Computer Magazine for Tandy® Computer Users

May 1986 Vol. III No. 11

Editor and Publisher
Lawrence C. Falk

Managing Editor Danny Humphress
Senior Editor Tamara Renee Dunn
Submissions Editor Jutta Kapfhammer
Reviews Editor E. Monica Dorth
Editorial Assistants Jo Anna Arnott, Wendy Falk, Judi Hutchinson, Angela Kapfhammer, Shirley Morgan
Technical Assistants Cray Augsburg, Chris Wehner
Contributing Editors William Barden, Jr., Robert D. Covington, John B. Harrell III
Consulting Editors Ed Ellers, Belinda C. Kirby, T. Kevin Nickols

Art Director Tracey Jones
Designers Jody Gilbert, Heidi Maxedon, Kevin Quiggins, Sandra Underwood
Production Assistant Cindy Jett

Chief of Typography Debbie Hartley
Typography Services
Jody Doyle, Suzanne Benish Kurowsky

Falsoft, Inc.

President Lawrence C. Falk

General Manager Patricia H. Hirsch
Asst. General Mgr. for Finance Donna Shuck
Admin. Asst. to the Publisher Sue E. Rodgers
PCMfest Coordinator Judy Brashear

Editorial Director James E. Reed
Asst. Editorial Director Jutta Kapfhammer
Creative Director Jerry McKiernan

Chief Bookkeeper Diane Moore
Advertising Accounts Beverly Taylor
Dealer Accounts Judy Quashnock
Accounts Payable/ADP Lisa Ragan
Director of Fulfillment Services Bonnie Frowenfeld
Asst. Dir. of Fulfillment Services Sandy Apple
Asst. Customer Service Manager Beverly Bearden
Word Processor Manager Patricia Eaton
Chief of Printing Services Melba Smith
Pre-press Production John Pike
Dispatch Janice Eastburn
Asst. Dispatch Mark Herndon
Business Assistants Laurie Falk, Doug Orr, Sharon Smith, Monica Wheat, Pam Workhoven

Advertising Coordinator Doris Taylor
Advertising Representative Kim Vincent
Advertising Assistant Debbie Baxter
(502) 228-4492
West Coast Advertising and Marketing Office
Director Cindy Shackleford
Advertising Representative Shirley Duranseau
For PCM Advertising and Marketing Office Information, see Page 130

PCM — The Personal Computer Magazine for Tandy® Computer Users is published every month of the year by FALSOFT, INC., The Falsoft Building, P.O. Box 385, Prospect, KY, 40059. Phone (502) 228-4492. **PCM — The Personal Computer Magazine for Tandy® Computer Users** and the **PCM** logotypes are registered ® trademarks of FALSOFT, Inc.

Second class postage paid Prospect, KY, and additional offices. USPS N 713-470 (ISSN 0747-0460). POSTMASTER: Send address changes to PCM, P.O. Box 385, Prospect, KY 40059. Forwarding Postage Guaranteed.

Subscriptions to **PCM — The Personal Computer Magazine for Tandy® Computer Users** are $28 per year in the United States. Canadian rates are U.S. $35. Surface mail to other countries is U.S. $64, air mail U.S. $85. All subscriptions begin with the next available issue.

Payment accepted by ViSA, MasterCard, American Express, Cash, Check or Money Order in United States currency only. Full refund after mailing of one issue. After two issues have been mailed, 10/12ths of the subscription price will be refunded. NO refund after mailing of three or more magazines.

Limited back issues are available. Please see notice for issues that are in print and their costs. Payment accepted by VISA, MasterCard, American Express, Cash, Check or Money Order in U.S. currency only.

Contents

Features

Departments

Reviews

Cover illustration copyright© 1986 by Fred Crawford

Tandy computers offer everything ...almost

Check out the software that completes their systems — *filePro*®

Tandy MS-DOS Software Comparison Chart

	pfs® File/Report	dBase™ II	*filePro* 16 Smallware
GENERAL CHARACTERISTICS:			
Menu driven	yes	no	YES ✓
Allows user to create integrated business systems	no	programmer required	YES ✓
Developed systems and data can be moved to multi-user environments	no	no	YES ✓
Professional support available from the software's authors	no	no	YES ✓
PRICE	$265	$595	$495
CAPACITIES:			
Fields per record	100	32	999 ✓
Characters per record	1679	1000	4608 ✓
Records per file	1300	65535	16,000,000 ✓
Indexes per file	1	7	12 ✓
Number of digits per numeric field	20	10	24 ✓
Number of files usable concurrently	1	2	10 ✓
Files span multiple drives	no	no	up to 8 ✓
FEATURES:			
Full-screen facility for creating custom screen layouts	yes	no	YES ✓
Full-screen facility for creating custom report layouts	no	no	YES ✓
Built-in field types (error checking)	no	3	12 ✓
User-defined field types	no	programmer required	200 ✓
Conditional math	no	programmer required	YES ✓
User-defined menus	no	programmer required	YES ✓
Change file layout without losing existing data	possible	possible	automatic ✓
Data protection	no	programmer required	YES ✓
Password security	no	programmer required	YES ✓

The new Tandy computers can really speed things up. If you don't let obsolete software bog you down.

Try *filePro* 16, the latest generation of data base software from the authors of Profile®.

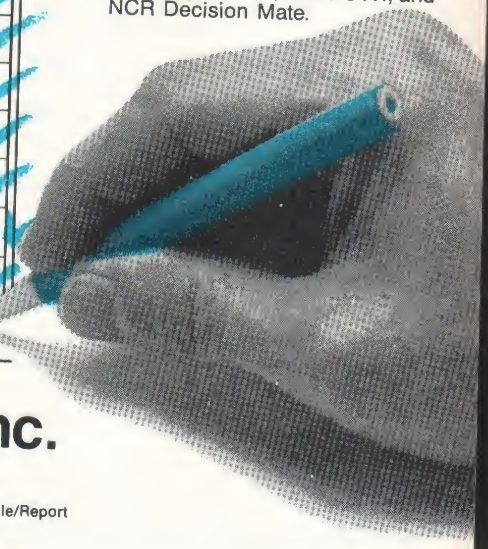
Choose it for its power (check our capacities); its ease-of-use (check our features); and its unique ability to grow as your needs grow, and even move the systems you create to multi-user computers.

filePro is available for the Tandy 1000, 1200 and 2000 computers (hard disk required). Available directly from us or through your Radio Shack store's Express Order Software service.

To order, call (800) 847-4740.

For customer service or when dialing in New York State, call **(212) 398-9290.** Ask for Smallware sales. Visa and MasterCard accepted.

Also available for other computers, including IBM PC, IBM PC-XT, and NCR Decision Mate.



The Small Computer Company, Inc.

240 **230 West 41st Street, Suite 1200, New York, New York 10036**

Riding the Leading Edge of a Growing Trend

How does it feel to be part of a growing trend?

If you're reading this magazine because you own one of Tandy's MS-DOS or portable computers, you're part of what appears to be the hottest trend in personal computing in the United States today. It is not just that you bought a computer — but that you bought a Tandy computer. Let's look at a few pieces of information (some of them from "sources," others from factual information):

The *Consumers Digest* Hall of Fame Award for meritorious service has been conferred on Tandy. What was it for?

Consumers Digest says it recognizes Tandy for "marketing its line of technologically advanced Tandy computers at competitive prices while maintaining one of the most extensive dealer/service networks in the country."

Well, we knew that, didn't we? But it certainly is nice that Tandy finally gets some recognition from someone as respectable as *Consumers Digest*.

To prove you are part of a really growing trend, all you have to do is to look at Tandy's reports to its stockholders. While many computer companies were reporting only so-so holiday seasons (or losses!), Tandy, overall, racked up its first-ever half-billion dollar month during December. That means its fourth quarter closed with its first $1 billion quarter in history.

How do I know this? Well, I have some Tandy stock. Sure am glad, too!

And you're *really* in the trend if you bought a Tandy MS-DOS computer! The way we hear it, although not officially, Tandy sold more than double the number of 1000s *alone* than IBM sold PCs. Tandy's 1000 sales also doubled Compaq's production, too, we hear.

And the newest computer from Tandy, the 3000? I understand that it is selling something like five times better than Tandy projected, while the 1200 is also said to be in healthy shape.

So, for that matter, is the Color Computer. Which also makes us glad here since we publish a Color Computer magazine, THE RAINBOW, as well.

What does all this mean to you? It means that more and more software and hardware people are paying more and more attention to the Tandy area. It means simple things like more attention to packaging drivers for your Tandy printers and modems and complicated ones like targeting sales and service efforts more toward the machine you own. These are important considerations. And we'll all reap the benefits of it.

When the University of Louisville here hired a big-name coach, Howard Schnellenberger, to run its football program, the school came up with the theme "Be Part of It!"

Aren't you glad you and your Tandy computer are "Part of It," too?

Despite being afflicted with the worst rainstorm in five years, PCMfest was a good experience in Palo Alto. A good number of firms had booths, the seminars were generally well-attended and there were a bunch of people on hand. Total for the whole show — PCMfest and RAINBOWfest — was a bit over 7,000.

What is traditionally our biggest RAINBOWfest (and I assume will be our biggest PCMfest, too) comes May 23-25 in Chicago. A registration form is located in this issue of PCM.

I hope you will not only attend, but will make your plans early. The hotel where the show is held, the Hyatt-Regency in Schaumburg, traditionally sells out for this one. So make your plans early and be with us.

And be even more a Part of It!

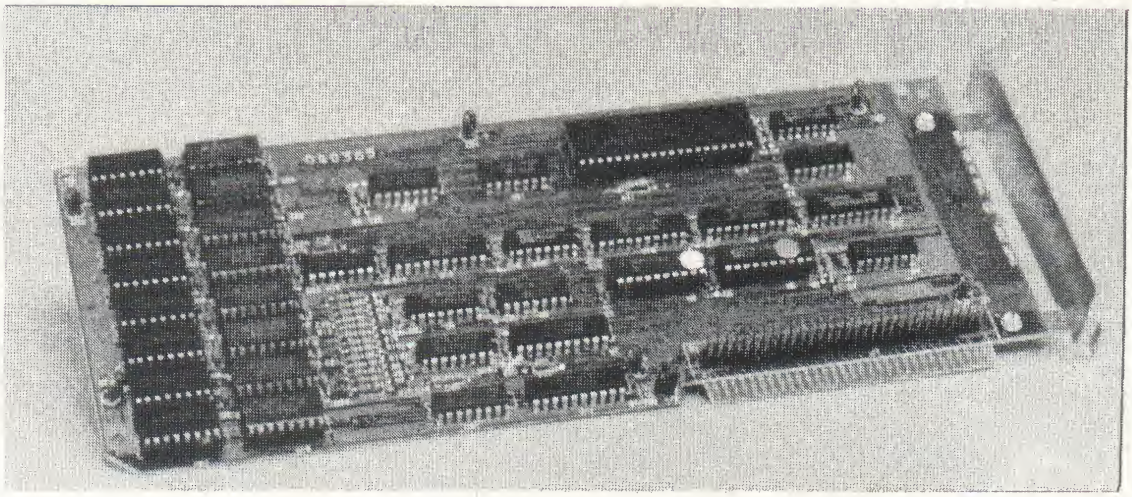
— Lonnie Falk

Tandy 1000 Memory Card

another high quality product from Southwestern Digital

Why spend a bundle on a Tandy board?
Our Board is only

$135.



Features:

- 512K of Memory
- DMA
- Expansion Port
- Gold Edge Cards
- Easy Installation
- 30 Day Satisfaction Guarantee Policy

The Southwestern Digital Memory Expansion Plus Card has all the features of the Radio Shack Board but the price; you save almost $400. Features include 512K installed, burned in, and tested to give you a total of 640K, a DMA circut that is fully tested for hard drive operation, and an expansion port that will work with any of the Radio Shack Memory Plus Expansion Card options. High quality manufacturing, and features such as gold plated card edges make this the logical choice in upgrading your memory.

Ordering Information
Call us or mail your order in. We accept Visa, Mastercard, and Certified Funds for quickest shipment. Personal checks are held for clearance. Add $5. for ground shipment, or $10 for UPS 2nd day air service. All products carry a 30 day satisfaction guarantee, and are warrantied for a full year.

Order Line
1-713-480-3296
Southwestern Digital
2515 West Main Street, Suite #337
Houston, Texas 77098

Mystery Socket

Editor:

I have been a reader of THE RAINBOW [for the Tandy Color Computer] for several years. When I purchased a Tandy 1000, I started looking for magazines to go with it. I was most pleased to find PCM.

I would, however, like to see more hardware oriented articles similar to "Turn of the Screw" and others in THE RAINBOW. This type of article would be a nice complement and would provide knowledge of the 1000 that seems to be non-existent.

As a case in point, when I started looking at the 1000 I inspected several at the local Radio Shack. The unit I purchased is listed as a 25-1000A. This suffix usually signifies a change in the product. Indeed, my machine has a 40-pin socket close to the 8088. This socket is non-existent on earlier 1000s I have inspected. I have read every article on the 1000 that I could find and never seen any mention of this socket. I have ordered the *Technical Reference Manual*, but I don't really think it will apply to the newer board.

I am wondering if you can offer any clarification of this matter. Could it even be a socket for an 8087 co-processor?

How about an article on using MS-DOS 3.1 on the 1000? This particular operating system seems to offer substantial improvements over the 2.1 series and is supposedly compatible. However, the $64,000 question is does it work?

Lionel P. Tercier
Bonnyville, Alberta

Editor's Note: It seems many of our contributors concentrate on software rather than hardware (hint, hint, hardware hackers, we need to hear from you). If you enjoy hardware articles, you'll find "Supercharging your Tandy 1000" in this issue very interesting. It shows how to increase your computer's speed by replacing the CPU with an inexpensive emulator chip.

Good guess on the mysterious empty socket in the new Tandy 1000s! It is, in fact, intended for the optional 8087 math co-processor. No official word from Tandy as yet on if and how owners of older machines can upgrade their machines to support the 8087.

A Better Blackboard

Editor:

I would like to compliment the staff of PCM for an excellent magazine. It is one of the only magazines that offers truly useful, productive and fun programs from which readers and amateur programmers can use and learn.

In Leonard Hyre's "The Electric Blackboard" (February 1986), there is a slight problem: `SCREEN 6` in Line 1940 will not allow `COLOR 12,0` to be used.

After running the program for my daughter a few times, I got annoyed with the idea of continually seeding the randomizer. I then borrowed a small addition from *Learning Basic for the Tandy 1000/2000* by David A. Lien. I added lines 3000 and 3010 as follows:

```
3000 RANDOMIZE VAL(RIGHT$
     (TIME$,2))
3010 RETURN
```

I also changed Lines 320 and 640 to `GOSUB 3000`. This way, the randomizer is seeded automatically by the computer's timer every time it is run.

John Sylvestre
Longueuil, Quebec

Update on Upgrading

Editor:

After reading Brian Alsop's article in the February 1986 issue ("Upgrading Your Tandy 1000"), I was so excited that I ran out and ordered a PBJ Board to replace my old Tandy memory board and a Qubie 20MB hard disk. Following Mr. Alsop's instructions, I eagerly installed both the memory board and the hard disk with little difficulty.

Unfortunately, that's when the troubles began. The hard disk formatted perfectly, however, every time I tried to read or write data to the disk, it would either hang up the computer or cause strange read/write errors. I also could not get the computer to boot directly from the hard disk.

After a few calls to Qubie, I discovered that Mr. Alsop didn't mention one important factor in his article: the BIOS. As I have an older Tandy 1000, it contains the older version of the BIOS (Version 01.00.00). It seems there is a bug in the old BIOS that causes the computer not to function correctly with a Western Digital hard disk controller board.

I dropped my machine off at Radio Shack and got a new BIOS chip installed (Version 01.01.00). I reinstalled the hard disk, and found that all my previous problems had disappeared!

The morale of the story: IF YOU ARE THINKING ABOUT UPGRADING TO A HARD DISK, MAKE SURE YOUR BIOS IS UPDATED TO VERSION 01.01.00! It only takes about a week to have Tandy install the chips, and it is relatively inexpensive — about $25. It will save you a lot of grief.

Qubie is also using a new EPROM chip for their Tandy 1000 controller boards. The old chips had "Tandy 1000" on the label; the new ones say "40-20007."

The hard disk has performed flawlessly, and is such a boon to computer productivity that it is almost like a miracle. Once you go to a hard disk you can *never* go to back to the floppy shuffle!

Thomas B. Simpson
Kalamazoo, MI

Powerful PCM

Editor:

Amidst the uncertainties of buying a personal and/or small business computer, I spent a great deal of time searching for the machine best suited to my needs and my financial limitations. I purchased a Tandy 1000 and have been unable to use it to its maximum. I find it quite powerful and more than able to satisfy my needs.

That brings me to PCM. I find it difficult to believe that each page is more informative than the last. Without a doubt, PCM is as powerful as my machine with respect to the fact it does for me just what I want it to do. Not one page has gone to waste. Other magazines twice the size cannot compare nor do they offer us the valuable information that you do.

Keep up the excellent work.

Stephen J. Devan
Methuen, MA

Praise for PBJ

Editor:

This is a letter of appreciation for the great and kind service given me by PBJ of Paterson, New Jersey. I bought an MFB-1000 memory expansion board and was uneasy about one of its features. In the course of discussing it with them by phone, they offered to let me bring the board in (they are close by).

We got friendly "VIP" attention and a complete rundown on the operation and features of this fine board. A user so often gets a rough time from the manufacturer that I feel it is important to emphasize a quality company like PBJ and their fine staff.

I'd like to pass along this tip on using the DB-25 port of the MFB-1000 in the RS-232 mode. These connections work with the Tandy DMP-110 printer:

DMP-100	DB-25	
1	No connection	
2	6	Status
3	7	Ground
4	2	Data

Jumper pins 4 and 8 at the DB-25

To set up the printer, at the `A>` prompt type:

```
MODE LPT1:=COM2:
MODE COM2: Baud rate, parity, word
length, stop bits
```

I use `MODE COM2:600,N,8,1`, though 1200 Baud is also available if the printer supports both.

R.E. Franklin
Westfield, NJ

Each and every month, PCM offers an assortment of programs for the home and office — utilities, business applications, games and more — all for the price of typing in the listings.

If you're a person on the move, and too busy to spend time typing lengthy listings, then PCM has just what you need — **PCM ON DISK.**

PCM ON DISK provides each issue's MS-DOS programs ready to load and run with only a few keystrokes!

AS AN ADDED BONUS, **PCM ON DISK** will include PCM's portable computer programs, which may be downloaded to portables for those readers who have both an MS-DOS machine and one of Tandy's portable computers.

That means you get a disk each month for a year, containing all programs listed in that month's issue, for an average cost of only $8.25!* We think it's worth your time to subscribe.

*Disks for individual issues (beginning this month) may be ordered for only $12 each.

PCM

PCM ON DISK
The Falsoft Building
P.O. Box 385
Prospect, KY 40059
(502) 228-4492

☐ Full Year's Subscription (specify starting month) _____

☐ Single Volume (specify month(s) & year) _____

Name _____

Address _____

City _____ State _____ ZIP _____

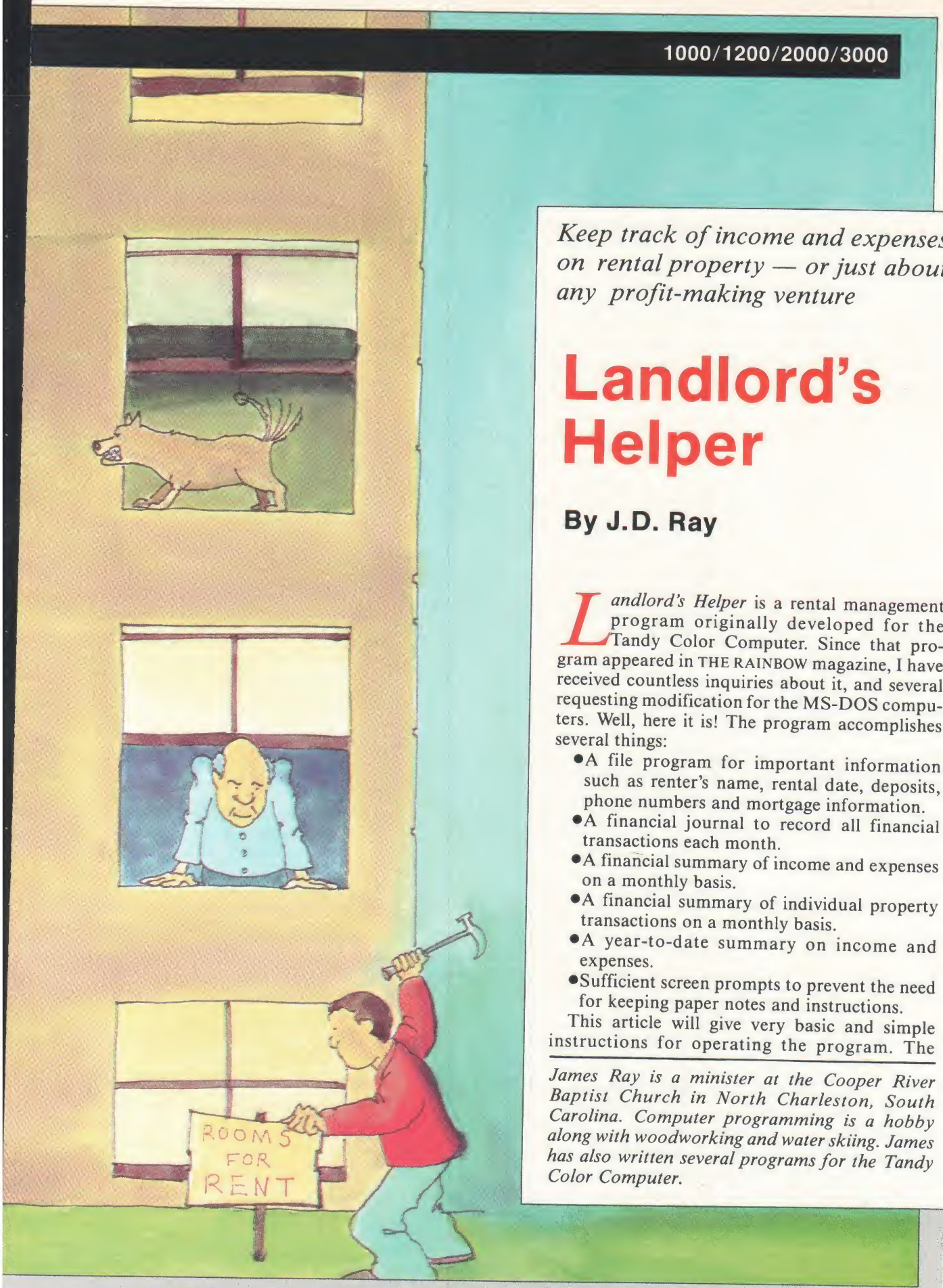
Payment Enclosed ☐

Charge: Visa ☐ MasterCard ☐ American Express ☐

My Account# _____ Card Exp. Date _____

Signature _____

*Subscriptions to PCM ON DISK are $99 a year in the United States. Canadian rate U.S. $115. All other countries U.S. $130; PCM ON DISK single copy rate is: $12 within the U.S.; U.S. $14, Canadian rate; U.S. $16, all other countries. Kentucky residents add 5% sales tax. All subscriptions begin with the current issue unless otherwise specified. Please allow 5 to 6 weeks for first copy. PCM ON DISK is not a stand-alone product. PCM magazine is a separate purchase. **U.S. currency *only*, please.** In order to hold down non-editorial costs, we do not bill.



Keep track of income and expenses on rental property — or just about any profit-making venture

Landlord's Helper

By J.D. Ray

Landlord's Helper is a rental management program originally developed for the Tandy Color Computer. Since that program appeared in THE RAINBOW magazine, I have received countless inquiries about it, and several requesting modification for the MS-DOS computers. Well, here it is! The program accomplishes several things:

- A file program for important information such as renter's name, rental date, deposits, phone numbers and mortgage information.
- A financial journal to record all financial transactions each month.
- A financial summary of income and expenses on a monthly basis.
- A financial summary of individual property transactions on a monthly basis.
- A year-to-date summary on income and expenses.
- Sufficient screen prompts to prevent the need for keeping paper notes and instructions.

This article will give very basic and simple instructions for operating the program. The

James Ray is a minister at the Cooper River Baptist Church in North Charleston, South Carolina. Computer programming is a hobby along with woodworking and water skiing. James has also written several programs for the Tandy Color Computer.

program is somewhat long, so these comments will be brief, but sufficient to use the program.

To Operate the Program

Assuming the program has been keyed in, load BASIC from your system disk and load LANDLORD. In order to use the program, you must:

1) Have LANDLORD loaded into memory from any drive (LOAD "*d*:LANDLORD" where *d*=drive letter).

2) Have a formatted file disk, or use main program disk, in Drive A for your working files.

Step one: From the main menu, select Item 4 to set up accounts. The purpose of this routine is to organize your property and accounts. Give this careful thought as this determines how the information will be organized. You can make corrections, additions and deletions, but it is better to have it right from the beginning. Organize your property accounts by assigning them a code. The code can be letters or numbers or both. The codes can be arranged by geographic location, property acquisition or value. For example, an apartment complex might be assigned Code A with the individual units by number: A01, A02, etc.

Income and expense accounts also need to be entered. You need to list these accounts after determining all of the possibilities. Consider every possible income item and every conceivable expense item. You are limited to 22 accounts each. The reason for the limitation is not memory or program design, but simply screen capacity. When developing the program, I opted for the program to prompt you with all accounts when needed. It would not hurt to set up several miscellaneous categories that can be changed later.

Step two: After setting up property, income and expense accounts, you are ready to enter specific property information. Select main menu Item 1. Follow the menu prompts to enter view or change data. To enter data, the screen prompts you for information. If any information does not apply to your situation, leave it blank by pressing ENTER. See Figure 1 for a screen summary.

Figure 1

```
                        Rental Property Information

          Property Code No:   E03                    Update: 11-21-1985
                   Address:   1504 White Circle      Record # 8
                      City:   Charleston
                     State:   SC
                  Zip Code:   29407

              Renter's Name:   Harold Goins
                Rental Date:   10/01/85
          Phone (XXX-XXXX):    322-3211
              Deposit Paid:      $300.00
              Monthly Rent:      $300.00

              Purchase Date:   01/01/85
            Purchase Price:     $24,000.00

Lien #1: Bankers Trust        Date Due (XX): 10   Monthly Mortgage:  $225.00
Lien #2:                      Date Due (XX):      Monthly Mortgage:    $0.00

                    [N]ext, [P]revious, [M]enu or [C]hange
```

Figure 2

```
                        Transactions for October     Prop. No.: All

Date   Tr #  Category   Received/Paid To:  For:                Prop:   Amount

10/01  1001   Rent       Joel Hanks         October Rent         A01   $250.00
10/01  1002   Rent       Mary Hobbs         October Rent         A02   $300.00
10/03  0001   Utilities  SCE&G              October Power Bill   A01   -$67.00
10/10  1003   Deposits   Joel Hanks         Rent Deposit         A01   $300.00
10/15  0004   Mortgage   First Federal      Oct. Mortgage        A01   -$195.00
10/10  1004   Rent       Steven Kingston    October Rent         C01   $350.00
10/15  0003   Plumbing   Johnny's Plumbing  Water Line Repair    A02   -$46.00


                        Ending Balance:  $1139.65
```

Figure 3

```
Year to Date Summmary - November
          10-31-1985
INCOME:
Rent                    $2,100.00
Refunds                     $0.00
Interest                    $0.00
Late Fees                   $0.00
Damages                     $0.00
Deposits                  $300.00
Misc.                       $0.00
Extra 1                     $0.00
Extra 2                     $0.00
    TOTAL INCOME:          -$745.00

EXPENSES:
Mortgage                 -$745.00
Utilities                 -$67.00
Plumbing                  -$46.00
Carpentry                   $0.00
Furniture                   $0.00
Yard Work                   $0.00
Gen. Main.                  $0.00
News. Ads.                  $0.00
Phone                       $0.00
Misc.                       $0.00
Extra 1                     $0.00
Extra 2                     $0.00
Extra 3                     $0.00
Extra 4                     $0.00
    TOTAL EXPENSES:        -$113.00

                          ==========
    BALANCE:              $1,542.00
```

Financial Data

Step three: To enter monthly transactions, select main menu Item 2. If you maintain a separate banking account for the rental property, simply enter these transactions from your checkbook and receipts. If you manage property through your personal bank account, simply go through your checkbook and enter those transactions that apply to your rental property. You can also list cash transactions.

After entering your transactions for the month, various reports can be printed based on these transactions. Transaction journals for the month can be printed by property location, or all transactions can be printed (to screen or printer). See Figure 2.

Step four: For monthly or year-to-date summaries, select main menu Item 3. This compiles the information listed in the transaction journal into useful summaries. Any of the following can be compiled to your screen or printer:

- An income summary listing all or selected income categories.
- An expense summary listing all or selected expense categories.
- A monthly summary of income and expenses.
- A year-to-date summary of income and expenses (Figure 3).

I hope you enjoy using *Landlord's Helper*. The program has many applications and is not limited to rental property. It can be used to handle financial information for any small business. I will be glad to assist anyone with specific applications for this program, and I am always interested if the program does not work as printed. Any input you can give that enhances the program would be appreciated and shared with interested persons. Feel free to contact me at 5065 France Avenue, North Charleston, SC 29406. Please enclose an SASE. □

The listing:

```
10 '***** LANDLORD"S HELPER *****
12 'COPYRIGHT- James Ray
14 CLEAR 9000:DIM COD$(25),ADR$(25),NAM$(25),EX$(23),IN$(25),MTH$(12),BAL(60),BZ
(50),BL(50),IT$(60),D(25),M(25),Q(25),Q1(25),PRI(25),LM$(25),PZ$(25)
16 K$="$$##,###.##":KK$="$$####.##":K2$="##"
18 FOR ZZ= 1 TO 12 : READ MTH$(ZZ): NEXT ZZ
20 C(1)=2:C(2)=10:C(3)=4:KEY OFF
22 DATA January,February, March , April ,  May  , June , July , August,September
,October,November,December
24 GOTO 940
26 CLS:COLOR C(1):LOCATE 7,29,0:PRINT"RENTAL PROPERTY MANAGEMENT"
28 LOCATE 9,34:PRINT"(C) by James Ray":FOR Z=1 TO 1000: NEXT Z
30 LOCATE 12,36:PRINT"MAIN MENU:":COLOR C(2)
32 LOCATE 13,30:PRINT"1) Property Information"
34 LOCATE 14,30:PRINT"2) List Monthly Transactions"
36 LOCATE 15,30:PRINT"3) Income/Expense Summary"
38 LOCATE 16,30:PRINT"4) Set-Up Accounts":LOCATE 17,30:PRINT"5) END Program"
40 LOCATE 19,30,0:PRINT"(1, 2, 3, 4, OR 5)":GOSUB 900
42 IF AT$<"0" OR AT$>"5" THEN 42
44 ON VAL(AT$) GOTO 742,270,406,50:STOP
46 '
48 '*************************************************************************
50 ' *** SET UP ACCOUNTS ***
52 '*************************************************************************
54 '
56 CLS:COLOR C(1):LOCATE 7,30:PRINT"FILE MAINTENANCE PROGRAM":COLOR C(2)
58 LOCATE 12,30:PRINT"Selections: ":LOCATE 13,30:PRINT"1) Property Codes"
60 LOCATE 14,30:PRINT"2) Expense Accounts":LOCATE 15,30:PRINT"3) Income Accounts
```

```
":LOCATE 16,3Ø:PRINT"4) END Job"
62 LOCATE 18,3Ø:PRINT"(1, 2, 3 or 4)":GOSUB 9ØØ
64 ON VAL(AT$) GOTO 66,144,2Ø2,9Ø4
66 'PROPERTY CODES
68 CLS:COLOR C(1):LOCATE 7,35:PRINT"Property Codes":COLOR C(2)
7Ø GOSUB 15Ø:ON VAL(AT$) GOSUB 72,94,1Ø6,2ØØ:GOTO 68
72 GOSUB 992:X=Ø:CLS:GOSUB 966
74 CLS:COLOR C(1):LOCATE 24,22:PRINT" Press <ENTER> When Finished":COLOR C(2)
76 X=X+1:L=L+1:IF L>23 THEN L=1
78 LOCATE L,1:COLOR C(1):PRINT"Property Code: ";:COLOR C(2):LINE INPUT COD$(X)
8Ø IF COD$(X)="" THEN CLOSE#1:GOTO 56
82 IF LEN(COD$(X))>3 THEN PRINT"  TOO LONG!! REDO ":GOTO 78
84 LOCATE L,2Ø:COLOR C(1):PRINT" Address: ";:COLOR C(2):LINE INPUT ADR$(X)
86 IF LEN(ADR$(X))>25 THEN PRINT"   TOO LONG!! REDO ":GOTO 84
88 LOCATE L,55:COLOR C(1):PRINT" Renter: ";:COLOR C(2):LINE INPUT NAM$(X)
9Ø IF LEN(NAM$(X))>2Ø THEN PRINT"   TOO LONG!! REDO ":GOTO 88
92 WRITE #1, CD$(X),ADR$(X),NAM$(X):GOTO 76
94 X=Ø:L=2:CLS:LOCATE 1,34:COLOR C(1):PRINT"PROPERTY CODES":GOSUB 968
96 IF EOF(1)=-1 THEN CLOSE#1:GOSUB 9ØØ:GOTO 68
98 X=X+1:L=L+1:LOCATE 2,1:COLOR C(1):PRINT"          Prop. Code   Renter's Name
    Address":COLOR C(2)
1ØØ INPUT#1,COD$(X),ADR$(X),NAM$(X)
1Ø2 LOCATE L,11:PRINT COD$(X):LOCATE L,22:PRINT NAM$(X):LOCATE L,43:PRINT ADR$(X
)
1Ø4 GOTO 96
1Ø6 X=Ø:CLS:GOSUB 968:GOSUB 97Ø
1Ø8 IF EOF(1)=-1 THEN 136
11Ø LOCATE 2,1:COLOR C(1):PRINT"Prop. Code   Renter's Name          Address":COLOR
 C(2)
112 X=X+1:INPUT#1,COD$(X),ADR$(X),NAM$(X)
114 LOCATE 3,3:PRINT SPACE$(4):LOCATE 3,14:PRINT SPACE$(21):LOCATE 3,35:PRINT SP
ACE$(25)
116 LOCATE 3,3:COLOR C(2):PRINT COD$(X):LOCATE 3,14:PRINT NAM$(X):LOCATE 3,35:PR
INT ADR$(X)
118 GOSUB 244:ON VAL(Q2$) GOSUB 12Ø,1Ø8,134:GOTO 134
12Ø CD$=COD$(X):LOCATE 3,3:INPUT COD$(X):IF COD$(X)="" THEN COD$(X)=CD$:LOCATE 2
Ø,1Ø:PRINT SPACE$(25)
122 IF COD$(X)="" THEN CLOSE#1:GOTO 56
124 IF LEN(COD$(X))>3 THEN LOCATE 2Ø,1Ø:COLOR C(3):PRINT"TOO LONG!! REDO ":COLOR
 C(2):GOTO 12Ø
126 NM$=NAM$(X):LOCATE 3,14:INPUT NAM$(X):IF NAM$(X)="" THEN NAM$(X)=NM$:LOCATE
2Ø,1Ø:PRINT SPACE$(2Ø)
128 IF LEN(ADR$(X))>2Ø THEN LOCATE 2Ø,1Ø:COLOR C(3):PRINT"TOO LONG!! REDO ":COLO
R C(2):GOTO 126
13Ø AR$=ADR$(X):LOCATE 3,35:INPUT ADR$(X):IF ADR$(X)="" THEN ADR$(X)=AR$:LOCATE
2Ø,1Ø:PRINT SPACE$(25)
132 IF LEN(NAM$(X))>25 THEN LOCATE 2Ø,1Ø:COLOR C(3):PRINT"TOO LONG!! REDO ":COLO
R C(2):GOTO 13Ø
134 WRITE #2,COD$(X),ADR$(X),NAM$(X):GOTO 1Ø8
136 PRINT:COLOR C(1):PRINT"Do you wish to add a Property (Y/N)";:COLOR C(2):INPU
T Q3$
138 IF Q3$= "N" OR Q3$= "n" THEN 142
14Ø X=X+1:CLS:COLOR C(1):LOCATE 2,1:PRINT"Prop. Code   Renter's Name          Addr
ess":COLOR C(2):GOTO 12Ø
142 CLOSE#2:CLOSE#1:GOTO 972
144 ' EXPENSE ACCOUNTS
146 CLS:COLOR C(1):LOCATE 1,34:PRINT"EXPENSE ACCOUNTS":COLOR C(2)
148 GOSUB 15Ø:ON VAL(AT$) GOSUB 156,17Ø,178,2ØØ:GOTO 146
15Ø LOCATE 12,3Ø:PRINT"Selections: ":LOCATE 13,3Ø:PRINT"1) Input Accounts"
```

```
152 LOCATE 14,3Ø:PRINT"2) View Accounts":LOCATE 15,3Ø:PRINT"3) Add/Change Accoun
ts":LOCATE 16,3Ø:PRINT"4) Return to Main Menu"
154 LOCATE 18,3Ø:PRINT"(1, 2, 3 or 4)":GOSUB 9ØØ:RETURN
156 GOSUB 992:CLS:X=Ø:GOSUB 976
158 CLS:LOCATE 24,22:COLOR C(1):PRINT"Press <ENTER> When Finished";:COLOR C(2)
16Ø X=X+1:LOCATE X,1:PRINT"Expense Account: #";:PRINT USING K2$;X;:LOCATE X,25:L
INE INPUT EX$(X):LOCATE X,4Ø:PRINT SPACE$(3Ø)
162 IF LEN(EX$(X))>1Ø THEN LOCATE X,4Ø:COLOR C(3):PRINT"Category is too Long! -R
edo!":COLOR C(2):LOCATE X,25:PRINT SPACE$(15):GOTO 16Ø
164 IF X=22 THEN CLOSE#1: GOTO 146
166 IF EX$(X)="" THEN CLOSE#1:GOTO 146
168 WRITE #1,EX$(X):GOTO 16Ø
17Ø X=Ø:CLS:LOCATE 1,34:COLOR C(1):PRINT"EXPENSE ACCOUNTS":COLOR C(2)
172 GOSUB 978
174 IF EOF(1)=-1 THEN CLOSE#1:GOSUB 9ØØ:GOTO 146
176 X=X+1:INPUT #1,EX$(X):COLOR C(1):PRINT "        Expense Account #";:PRINT US
ING K2$;X;:COLOR C(2):PRINT"    ";EX$(X):GOTO 174
178 CLS:X=Ø:GOSUB 978:GOSUB 97Ø
18Ø IF EOF(1)=-1 THEN 192
182 X=X+1:INPUT #1,EX$(X)
184 CLS:LOCATE 2,1Ø:COLOR C(1):PRINT "Expense Account #";X;"    ";:COLOR C(2):PRI
NT EX$(X)
186 GOSUB 244:ON VAL(Q2$) GOSUB 188,18Ø,19Ø:GOTO 19Ø
188 COLOR C(1):PRINT:PRINT"  New Account: ";:COLOR C(2):INPUT EX$(X)
19Ø WRITE#2,EX$(X):GOTO 18Ø
192 PRINT:COLOR C(1):PRINT"Do you wish to add an Account (Y/N)";:COLOR C(2):INPU
T Q3$
194 IF Q3$="N" OR Q3$= "n" THEN 198
196 X=X+1:IF X=22 THEN 198:GOTO 188
198 CLOSE#2,#1:GOSUB 98Ø
2ØØ CLS:GOTO 56
2Ø2 ' INCOME ACCOUNTS
2Ø4 CLS:COLOR C(1):LOCATE 1,34:PRINT"INCOME ACCOUNTS":COLOR C(2)
2Ø6 GOSUB 15Ø:ON VAL(AT$) GOSUB 2Ø8,224,234,2ØØ:GOTO 2Ø4
2Ø8 GOSUB 992:CLS:X=Ø:GOSUB 984
21Ø LOCATE 24,22:COLOR C(1):PRINT"Press <ENTER> when Finished";:COLOR C(2)
212 X=X+1
214 LOCATE X,1:PRINT"Income Account: #";:PRINT USING K2$;X;:LOCATE X,25:LINE INP
UT IN$(X):LOCATE X,4Ø:PRINT SPACE$(3Ø)
216 IF LEN(IN$(X))>1Ø THEN LOCATE X, 4Ø:COLOR C(3):PRINT"Category is too Long! -
REDO!":COLOR C(2):LOCATE X,25:PRINT SPACE$(15):GOTO 214
218 IF X=22 THEN CLOSE#1:GOTO 2Ø4
22Ø IF IN$(X)="" THEN CLOSE #1:GOTO 2Ø4
222 WRITE #1, IN$(X):GOTO 212
224 CLS:X=Ø:LOCATE 1,34:COLOR C(1):PRINT"INCOME ACCOUNTS":COLOR C(2)
226 GOSUB 986
228 IF EOF(1)=-1 THEN CLOSE#1:GOSUB 9ØØ:GOTO 2Ø4
23Ø X=X+1:INPUT#1,IN$(X)
232 COLOR C(1):PRINT"        Income Account #";:PRINT USING K2$;X;:COLOR C(2):PR
INT"    ";IN$(X):GOTO 228
234 CLS:X=Ø:GOSUB 986:GOSUB 97Ø
236 IF EOF(1)=-1 THEN 256
238 X=X+1:INPUT #1,IN$(X)
24Ø CLS:COLOR C(1):LOCATE 2,1Ø:PRINT "Income Account #";X;"    ";:COLOR C(2):PRIN
T IN$(X)
242 GOSUB 244:ON VAL(Q2$) GOSUB 252,236,254:GOTO 254
244 PRINT:COLOR C(1):PRINT" Do you wish to: ":PRINT"  1) Change the account"
246 PRINT"  2) Delete the account":PRINT"  3) See next account":COLOR C(2)
248 Q2$=INKEY$:IF Q2$="" THEN 248:RETURN
```

```
25Ø RETURN
252 COLOR C(1):PRINT"  New Account: ";:COLOR C(2):INPUT IN$(X)
254 WRITE #2,IN$(X):GOTO 236
256 COLOR C(1):PRINT"Do you wish to add an Account   (Y/N)";:COLOR C(2):INPUT Q3
$
258 IF Q3$="N" OR Q3$= "n" THEN 262
26Ø X=X+1:IF X=23 THEN 262:GOTO 252
262 CLOSE#2,#1
264 GOTO 988
266 '
268 '*************************************************************************
27Ø ' *** LIST MONTHLY TRANSACTION$ ***
272 '*************************************************************************
274 '
276 'SUBROUTINE FOR INPUTTING MONTHLY INFORMATION
278 CLS:COLOR C(1):LOCATE 7,3Ø:PRINT"MONTHLY INCOME & EXPENSES":COLOR C(2)
28Ø MN$="" :BAL=Ø:LOCATE 9,3Ø:PRINT"Selections:"
282 LOCATE 1Ø,3Ø:PRINT"1) Input Transactions":LOCATE 11,3Ø:PRINT"2) View Data"
284 LOCATE 12,3Ø:PRINT"3) END Job":LOCATE 14,3Ø:PRINT"(1, 2, or 3)":GOSUB 9ØØ
286 ON VAL(AT$) GOTO 288, 352, 9Ø4:GOTO 278
288 CLS:COLOR C(1):LOCATE 9,32:PRINT"MONTHLY JOURNAL"
29Ø LOCATE 11,1Ø:INPUT"Month (Ø1-12): ";M1$:IF M1$<"Ø1" OR M1$>"12" THEN 29Ø
292 LOCATE 12,7:INPUT"Previous Balance: ";PB:COLOR C(2):GOSUB 898:TR=LOF(1)/7Ø
294 CLS:TR=TR+1:COLOR C(1):LOCATE 7,1Ø:M1=VAL(M1$):PRINT MTH$(M1);:PRINT " TRANS
ACTIONS":LOCATE 8,11:PRINT"Transaction # ";TR:LOCATE 1Ø,1:PRINT"Expense or Incom
e (E/I): ?";
296 AN$= INKEY$ :IF AN$="" THEN 296
298 IF AN$="I" OR AN$="i" THEN LOCATE 1Ø,28:PRINT"Income":GOTO 32Ø
```

```
3ØØ IF AN$="E" OR AN$="e" THEN LOCATE 1Ø,28:PRINT"Expense":GOTO 3Ø2 ELSE 294
3Ø2 COLOR C(1):LOCATE 11,4:PRINT"Expense Date (MM/DD): ":LOCATE 12,8:PRINT"Check
 No. (XXXX): ":LOCATE 13,17:PRINT"Paid To: "
3Ø4 LOCATE 14,12:PRINT"For What (#): ":LOCATE 15,12:PRINT"Property No.: ":LOCATE
 16,13:PRINT"Description: ":LOCATE 17,18:PRINT"Amount: ":LOCATE 19,17:PRINT"Bala
nce: "
3Ø6 COLOR C(2):LOCATE 11,26:INPUT D1$:LOCATE 12,26:INPUT D2$:LOCATE 13,26:INPUT
IC$
3Ø8 GOSUB 916:LOCATE 14,26:INPUT F
31Ø IF F>Y THEN 3Ø8
312 F$=EX$(F):Y=Ø
314 LOCATE 15,26:INPUT P$:LOCATE 16,28,1:LINE INPUT EC$:LOCATE 17,26:INPUT AMT
316 TAM=AMT : TAM=-TAM: AMT=-AMT :PB=PB+TAM :LOCATE 19,26:PRINT USING K$;PB
318 GOTO 338
32Ø COLOR C(1):LOCATE 11,7:PRINT"Dep. Date (MM/DD): ":LOCATE 12,9:PRINT"Dep. No.
 (XXXX): ":LOCATE 13,13:PRINT"Income From: ":LOCATE 14,11:PRINT"Income For (#) "
:LOCATE 15,12:PRINT"Property No.: ":LOCATE 16,13:PRINT"Description: ? "
322 LOCATE 17,7:PRINT"Amount of Deposit: ":LOCATE 19,17:PRINT"Balance: ":COLOR C
(2)
324 LOCATE 11,26:INPUT D1$:LOCATE 12,26:INPUT D2$:LOCATE 13,26:INPUT IC$
326 GOSUB 926:LOCATE 14,26:INPUT F
328 IF F>Y THEN 326
33Ø F$=IN$(F):Y=Ø
332 LOCATE 15,26:INPUT P$:LOCATE 16,28,1:LINE INPUT EC$:LOCATE 17,26:INPUT AMT
334 TAM= AMT : PB=PB + TAM
336 LOCATE 19,26:PRINT USING K$;PB
338 LOCATE 21,5:INPUT"Is This Information Correct (Y/N)? ";AC$
34Ø IF AC$="N" OR AC$="n" THEN 294 ELSE 342
342 LSET COD$=P$:LSET CAT$=F$:LSET DT$=D1$:LSET DL$=D2$:LSET PDTO$=IC$:LSET DSC$
=EC$:LSET AMT$=MKD$(AMT)
344 PUT #1,TR
346 LOCATE 23,2Ø:INPUT"More Input (Y/N)? ";AN$
348 IF AN$="Y" OR AN$="y" THEN 294 ELSE 35Ø
35Ø LOCATE 23,4Ø:PRINT "SAVING";TR;:PRINT " TRANSACTIONS":CLOSE#1:RETURN
352 'SUB-ROUTINE FOR VIEWING TRANSACTIONS
354 CLS:BAL=Ø:LOCATE 1,3Ø:COLOR C(1):PRINT"TRANSACTION JOURNAL"
356 PT$="":LOCATE 3,6:PRINT"What Month (Ø1-12): ";:COLOR C(2):INPUT M1$:LOCATE 4
,8:COLOR C(1):PRINT"Previous Balance: ";:COLOR C(2):INPUT BAL:BX=BAL:GOSUB 898:M
1=VAL(M1$)
358 CLS:COLOR C(1):LOCATE 4,1Ø:PRINT"Do You Want to See All Tranactions or Just
One Property Location? "
36Ø LOCATE 5,1Ø:PRINT"All or One (A/O): ";:COLOR C(2):INPUT XX$:IF XX$="A" OR XX
$="a" THEN MN$="All"
362 IF XX$="O" OR XX$= "o" THEN LOCATE 7,1Ø:COLOR C(1):PRINT"What is the Propert
y No. (XXX): ";:COLOR C(2):INPUT MN$
364 COLOR C(1):LOCATE 9,1Ø:PRINT"Send Data to Printer (Y/N)";:COLOR C(2):INPUT P
T$:CLS:Y=3:TR=1
366 LOCATE 1,25:COLOR C(1):PRINT"TRANSACTIONS FOR MONTH: ";MTH$(M1):LOCATE 1,6Ø:
PRINT"Prop. No.: ";:COLOR C(2):PRINT MN$
368 IF PT$="Y" OR PT$="y" THEN GOSUB 666:LPRINT TAB(3Ø) "Transactions for " MTH$
(M1) "     Prop. No.: ";MN$
37Ø COLOR C(1):LOCATE 3,1:PRINT"Date   Tr #  Category  Received/Paid To:  For:
            Prop:    Amount":COLOR C(2)
372 IF PT$="Y" OR PT$="y" THEN GOSUB 666:LPRINT"   Date   Tr #  Category  Receiv
ed/Paid To:  For:                Prop:    Amount":LPRINT ""
374 GET #1,TR:AMT=CVD(AMT$):IF EOF(1)=-1 THEN CLOSE#1:GOTO 394
376 IF XX$="A" OR XX$="a" THEN 382
378 IF XX$="O" AND COD$ <> MN$ THEN 392
38Ø IF XX$="o" AND COD$ <> MN$ THEN 392
```

Take off with a Tandy® portable and have real computing power wherever you travel.

Great traveling companions

Whether you're a businessperson, journalist, police officer or student, you can't afford idle time. Our affordable Model 100, Tandy 200, and Tandy 600 portable computers are invaluable tools that help keep you on top of your work wherever you go.

Computing power for less

The popular Model 100 (26-3802) is perfect for writing letters, reports and other correspondence anywhere, at any time. Or use the 100 as your personal appointment calendar and name/address and phone directory. This lightweight computer was the first to feature five instant-on programs, as well as a 40×8 display—all built in.

At only $499, this powerful little traveler offers you more options than you thought possible—including a built-in modem and a communications program that lets you access other computers or national information services by phone. You can even write your own programs in BASIC.

Powerful portability

The $799 Tandy 200 represents a true second generation in portable computers. Not only does it come with its own built-in software, improved features, a bigger 40×16 fliptop screen, and a larger memory, but it weighs only 4½ pounds.

Plus, the Tandy 200 (26-3860) comes with powerful Multiplan® spreadsheet analysis built right into the memory. You can do complex spreadsheet calculations at the touch of a key. Answer thousands of "what if . . . ?" questions instantly.

Our most advanced laptop

The high-performance Tandy 600 (26-3901) features a 16-bit microprocessor, an 80×16 display, a built-in 3½" disk drive that stores 360K of data, and 32K RAM (expandable to 224K).

At $1599, the Tandy 600 also includes five resident applications—all designed by Microsoft for anyone who's always on the go. With the Tandy 600's larger display and expanded memory, Multiplan can handle more information.

Using MS-Word®, you'll also have quicker access to documents and more storage capacity with the built-in disk drive. File is an electronic database for names and addresses, expenses, client billing and more. With the Telecom program and the Tandy 600's built-in modem, you can keep in touch over phone lines.

Don't leave the office without one

Tandy offers you a choice of three portable computers—one is sure to fit your needs. See them all at your local Radio Shack Computer Center!

Available at over 1200 Radio Shack Computer Centers and at participating Radio Shack stores and dealers.

Radio Shack®
COMPUTER CENTERS
A DIVISION OF TANDY CORPORATION

Send me a free Computer Catalog.

Radio Shack, Dept. 86-A-900
300 One Tandy Center
Fort Worth, Texas 76102

Name _____
Address _____
City _____
State _____
ZIP _____
Phone _____

Prices apply at Radio Shack Computer Centers and participating stores and dealers. Multiplan and MS-Word/Registered TM Microsoft Corp.


TANDY 200

```
382 M1=VAL(M1$):BAL=BAL+AMT
384 Y=Y+1:COLOR C(2):LOCATE Y,1:PRINT DT$;:LOCATE Y,8:PRINT DL$;:LOCATE Y,14:PRI
NT CAT$;:LOCATE Y,24:PRINT PDTO$;:LOCATE Y,43:PRINT DSC$
386 LOCATE Y,63:PRINT COD$;:LOCATE Y,68:PRINT USING KK$;AMT
388 IF PT$="Y" OR PT$="y" THEN LPRINT "   " DT$ "  " DL$ "  " CAT$ PDTO$ "  " DS
C$ COD$ "  ";:LPRINT USING KK$; AMT
39Ø IF Y=23 THEN Y=2:GOSUB 4ØØ
392 TR=TR+1:GOTO 374
394 CLOSE#1:Y=Y+2:LOCATE Y,28:COLOR C(1):PRINT"Ending Balance: ";:COLOR C(2):PRI
NT USING KK$;BAL
396 IF PT$="Y" OR PT$="y" THEN GOSUB 666:LPRINT TAB(3Ø) "Ending Balance: ";:LPRI
NT USING KK$;BAL:GOSUB 666
398 GOSUB 9ØØ:GOTO 278
4ØØ LOCATE 24,27:COLOR C(1):PRINT"Hit <ENTER> to Continue";:COLOR C(2):GOSUB 9ØØ
:GOSUB 9Ø4
4Ø2 '
4Ø4 '*************************************************************************
4Ø6 ' *** INCOME/EXPENSE SUMMARY ***
4Ø8 '*************************************************************************
41Ø '
412 CLS:PT$="Q":BAL=Ø:COLOR C(1):LOCATE 7,3Ø:PRINT"INCOME/EXPENSE SUMMARY"
414 COLOR C(2):LOCATE 9,3Ø:PRINT"Selections: "
416 LOCATE 1Ø,3Ø:PRINT"1) Expense Summary"
418 LOCATE 11,3Ø:PRINT"2) Income Summary"
42Ø LOCATE 12,3Ø:PRINT"3) Monthly Summary"
422 LOCATE 13,3Ø:PRINT"4) Year-to-Date Summary"
424 LOCATE 14,3Ø:PRINT"5) END Job":LOCATE 16,3Ø:PRINT"(1, 2, 3, 4 or 5)":GOSUB 9
ØØ
426 IF AT$<"Ø" OR AT$>"5" THEN 412
428 ON VAL(AT$) GOTO 432,478,53Ø,672,9Ø4:GOTO 412
43Ø 'ROUTINE TO VIEW EXPENSES
432 CLS:BAL=Ø:LOCATE 9,32:COLOR C(1):PRINT"Expense Summary":LOCATE 11,2Ø:PRINT"W
hat Month (Ø1-12): ";:COLOR C(2):INPUT M1$:M1=VAL(M1$)
434 LOCATE 13,2:COLOR C(1):PRINT"All Categories or One Expense Category? "
436 GOSUB 916
438 LOCATE 14,24:COLOR C(1):PRINT"Category (ALL/#)";:COLOR C(2):INPUT F$
44Ø IF F$="ALL" THEN CT$="ALL":GOTO 444
442 F=VAL(F$):CT$=EX$(F):L1=LEN(CT$)
444 GOSUB 898:CLS:COLOR C(1):LOCATE 1,3Ø:PRINT MTH$(M1);" Expenses":LOCATE 1,55:
PRINT" Category - ";:COLOR C(2):PRINT CT$
446 TR=1:Y=3:BAL=Ø
448 COLOR C(1):LOCATE 3,1:PRINT"Date   Ck #  Category  Paid To:           For:
          Prop:    Amount":COLOR C(2)
45Ø GET #1,TR:CZ$=LEFT$(CAT$,L1):IF EOF(1)=-1 THEN CLOSE#1:GOTO 466
452 IF CT$="ALL" THEN 456
454 IF CT$=CZ$ THEN 456 ELSE 464
456 AMT=CVD(AMT$):IF AMT > Ø THEN 464
458 BAL=BAL+AMT
46Ø Y=Y+1:COLOR C(2):LOCATE Y,1:PRINT DT$;:LOCATE Y,8:PRINT DL$;:LOCATE Y,14:PRI
NT CAT$;:LOCATE Y,24:PRINT PDTO$;:LOCATE Y,43:PRINT DSC$
462 LOCATE Y,63:PRINT COD$;:LOCATE Y,68:PRINT USING KK$;AMT
464 TR=TR+1:GOTO 45Ø:CLOSE#1
466 Y=Y+2:LOCATE Y,28:COLOR C(1):PRINT"Total Expenses: ";:COLOR C(2):PRINT USING
 KK$;BAL:GOSUB 9ØØ
468 LOCATE 24,3Ø:COLOR C(1):PRINT"<A>nother or <M>enu";:COLOR C(2)
47Ø YY$=INKEY$:IF YY$="" THEN 47Ø
472 IF YY$="A" OR YY$="a" THEN 432
474 IF YY$="M" OR YY$="m" THEN 412
476 CLOSE#1:RETURN
```

```
478 'SUBROUTINE TO VIEW INCOME
480 CLS:BAL=0:LOCATE 9,32:COLOR C(1):PRINT"Income Summary":LOCATE 11,20:PRINT"Wh
at Month (01-12): ";:COLOR C(2):INPUT M1$:M1=VAL(M1$)
482 LOCATE 13,3:COLOR C(1):PRINT"All Categories or One Income Category? "
484 GOSUB 926
486 LOCATE 14,23:COLOR C(1):PRINT"Category (ALL/#) ";:COLOR C(2):INPUT F$
488 IF F$="ALL" THEN CT$="ALL":GOTO 492
490 F=VAL(F$):CT$=IN$(F):L1=LEN(CT$)
492 GOSUB 898
494 CLS:COLOR C(1):LOCATE 1,30:PRINT MTH$(M1);" Income":LOCATE 1,55:PRINT"Catego
ry: ";:COLOR C(2):PRINT CT$
496 TR=1:Y=3:BAL=0
498 COLOR C(1):LOCATE 3,1:PRINT"Date   Dep#  Category  Received From      For:
          Prop:    Amount":COLOR C(2)
500 GET #1,TR:CZ$=LEFT$(CAT$,L1):IF CT$="All" THEN 506
502 IF EOF(1)=-1 THEN CLOSE #1:GOTO 514
504 IF CT$=CZ$ THEN 506 ELSE 512
506 AMT=CVD(AMT$):IF AMT < 0 THEN 512
508 Y=Y+1:COLOR C(2):LOCATE Y,1:PRINT DT$;:LOCATE Y,8:PRINT DL$;:LOCATE Y,14:PRI
NT CAT$;:LOCATE Y,24:PRINT PDTO$;:LOCATE Y,43:PRINT DSC$
510 LOCATE Y,63:PRINT COD$;:LOCATE Y,68:PRINT USING KK$;AMT:BAL=BAL+AMT
512 TR=TR+1:GOTO 500:CLOSE #1
514 Y=Y+2:LOCATE Y,28:COLOR C(1):PRINT"Total Income: ";:COLOR C(2):PRINT USING K
K$;BAL:GOSUB 900
516 LOCATE 24,30:COLOR C(1):PRINT"<A>nother or <M>enu";:COLOR C(2)
518 YY$=INKEY$:IF YY$="" THEN 518
520 IF YY$="A" OR YY$="a" THEN 480
522 IF YY$="M" OR YY$="m" THEN 412
524 CLOSE#1:RETURNRUN
526 '
528 '*************************************************************************
530 '  *** MONTHLY SUMMARY ***
532 '*************************************************************************
534 '
536 CLS:LOCATE 9,32:COLOR C(1):PRINT"Monthly Summary":LOCATE 11,20:PRINT"What Mo
nth (01-12)? ";:COLOR C(2):INPUT M1$:M1=VAL(M1$)
538 COLOR C(1):LOCATE 15,20:PRINT"SEND DATA TO PRINTER (Y/N) ";:COLOR C(2):INPUT
 PT$
540 CLS:BAL=0:AMT=0:Y=2:R1=0:T1=0:TT=0:X=1:T=1
542 IF PT$="Y" OR PT$="y" THEN GOSUB 648
544 T1=0:LOCATE 1,30:COLOR C(1):PRINT MTH$(M1);" Monthly Summary":COLOR C(2)
546 GOSUB 898:BAL(X)=0:AMT=0
548 IF R1=0 THEN GOSUB 590 ELSE 592
550 L1=LEN(CT$):TR=1
552 GET#1,TR
554 IF EOF(1)=-1 THEN CLOSE#1: GOTO 564
556 AMT=CVD(AMT$):CZ$=LEFT$(CAT$,L1)
558 IF CT$=CZ$ THEN 560 ELSE 562
560 BAL(X)=BAL(X)+AMT
562 TR=TR+1:GOTO 552
564 Y=Y+1:LOCATE Y,29:PRINT CT$;:LOCATE Y,43:PRINT USING K$;BAL(X):T1=T1+BAL(X)
566 IF Y>22 THEN GOSUB 736:Y=2:CLS
568 IF PT$="QX" THEN GOSUB 652
570 X=X+1:T=T+1
572 IF R1=1 THEN 576
574 IF IN$(T)="" THEN 580 ELSE 546
576 IF EX$(T)="" THEN 594 ELSE 546
578 IF R1=1 THEN GOTO 594
580 Y=Y+2:LOCATE Y,29:COLOR C(1):PRINT"Total Income:    ";:COLOR C(2):LOCATE Y,43
```

```
:PRINT USING K$;T1:Y=Y+1
582 IF Y>22 THEN GOSUB 736:Y=2
584 IF PT$="QX" THEN GOSUB 658
586 R1=1:IN(1)=T1:T=1
588 CLOSE#1:GOTO 544
59Ø CT$=IN$(T):RETURN
592 CT$=EX$(T):GOTO 55Ø
594 Y=Y+2:LOCATE Y,29:COLOR C(1):PRINT"Total Expenses: ";:COLOR C(2):LOCATE Y,43
:PRINT USING K$;T1
596 IF Y>22 THEN GOSUB 736:Y=2
598 TT=IN(1)+T1:IF PT$="QX" THEN GOSUB 66Ø
6ØØ Y=Y+2:LOCATE Y,29:COLOR C(1):PRINT"Balance: ";:COLOR C(2):LOCATE Y,43:PRINT
USING K$;TT
6Ø2 IF PT$="QX" THEN 662
6Ø4 CLOSE#1:Y=1:X=1:R=Ø
6Ø6 GOSUB 9ØØ
6Ø8 CLS:COLOR C(1):LOCATE 1Ø,18:PRINT"Do You Want to Update the Yearly Summary F
ile?":LOCATE 12,28:PRINT"Update Only One Time!"
61Ø LOCATE 14,25:PRINT"(<Y>es/<N>o/<F>irst Time) ";:COLOR C(2):INPUT UP$
612 IF UP$="" THEN 612
614 IF UP$="Y" OR UP$="y" THEN 618
616 IF UP$="F" OR UP$="f" THEN 626 ELSE 412
618 CLS:Z=Ø:COLOR C(1):LOCATE 7,3Ø:PRINT"Loading Previous Summary File":T=Ø:Z=Ø
62Ø GOSUB 936
622 IF EOF(1)=-1 THEN CLOSE#1:GOTO 626
624 Z=Z+1:INPUT#1,IT$(Z),BL(Z):GOTO 622
626 GOSUB 938
628 CLS:LOCATE 7,3Ø:PRINT"Adding Current Month":R=Ø:W=1:T=1
63Ø IF R=Ø THEN BZ(W)=BAL(T)+BL(T):WRITE #1,IN$(W),BZ(W)
632 IF R=1 THEN BZ(W)=BAL(T)+BL(T) :WRITE#1,EX$(W),BZ(W)
634 W=W+1:T=T+1
636 IF R=Ø THEN GOTO 642
638 IF R=1 THEN GOTO 644
64Ø W=1:R=1:GOTO 632
642 IF IN$(W)="" THEN GOTO 64Ø ELSE 63Ø
644 IF EX$(W)="" THEN GOTO 646 ELSE 632
646 CLOSE#1:LOCATE 24,3Ø:PRINT"Hit <ENTER> to Continue":GOSUB 9ØØ:GOTO 412
648 GOSUB 666:LPRINT TAB(36) MTH$(M1);" Monthly Summary":LPRINT TAB(41) DATE$
65Ø GOSUB 666:PT$="QX":LPRINT TAB(3Ø) "INCOME":RETURN
652 LPRINT TAB(3Ø) CT$;"          ";
654 P1=11-L1:FOR V=Ø TO P1:LPRINT " ";:NEXT V
656 LPRINT USING K$;BAL(X):RETURN
658 LPRINT TAB(51) "__________":LPRINT TAB(34) "TOTAL           ";:LPRINT USING
K$;T1:LPRINT "":LPRINT TAB(3Ø) "EXPENSES: ":RETURN
66Ø LPRINT TAB(51) "__________":LPRINT TAB(34);"TOTAL EXPENSES: ";:LPRINT USING
K$;T1:GOSUB 666:RETURN
662 LPRINT TAB(51) "==========":LPRINT TAB(34) "BALANCE:         ";:LPRINT USING
K$;TT:GOSUB 666:GOSUB 9ØØ:GOTO 412
664 PT$="Q":CLOSE#1:GOTO 6Ø8
666 FOR X=1 TO 2:LPRINT" ":NEXT X:RETURN
668 '
67Ø '*************************************************************************
672 ' *** YEAR TO DATE SUMMARY ***
674 '*************************************************************************
676 '
678 CLS:COLOR C(1):LOCATE 1Ø,1Ø:PRINT"You must summarize current month and add t
o Yearly Summary File                before Year-to-Date File will be accurate!
":PRINT"Continue (Y/N) ";:COLOR C(2):INPUT Q$:PT$="q":TT=Ø
68Ø IF Q$="" THEN 68Ø
```

Memory Upgrades for Tandy Portables!

Model 600

96K RAM Upgrade
$249. or $465. for two

Save up to **$335.** with this upgrade designed specifically for the Model 600. The 96K upgrade card fits into existing slots in the Model 600 and adds either 96K or 192K of RAM to the computer.

Model 200

24K RAM Upgrade
$89.00
Two for $170.

Save up to **$330.** with this exact replacement while bringing your Model 200 up to 72K.

Model 100, 8201 or M10

8K RAM Upgrade
$27.00
$75. for pkg. of 3

Save up to $282. upgrading an 8K Model 100 to 32K. The modules also fit the NEC 8201 and the Olivetti M10.

★ ★ ★ BUY QUALITY & SAVE ★ ★ ★

All three upgrades feature low power version of CMOS static RAM for minimum battery drain and maximum reliability. Our products are 100% tested before they leave the factory. Every upgrade carries with it a 30 day satisfaction money back guarantee and 1 yr. replacement warranty. Step by step illustrated instructions are included, and our customer service support is as close as your phone.

TO ORDER
CALL 714-540-1174
or WRITE
M/C, VISA, CHECK or MONEY ORDER
CALIF. RESIDENTS ADD 6% Sales Tax
SHIPPING-ADD
UPS Ground $1.50
2nd Day Air $4.00
Next Day Air $12.00

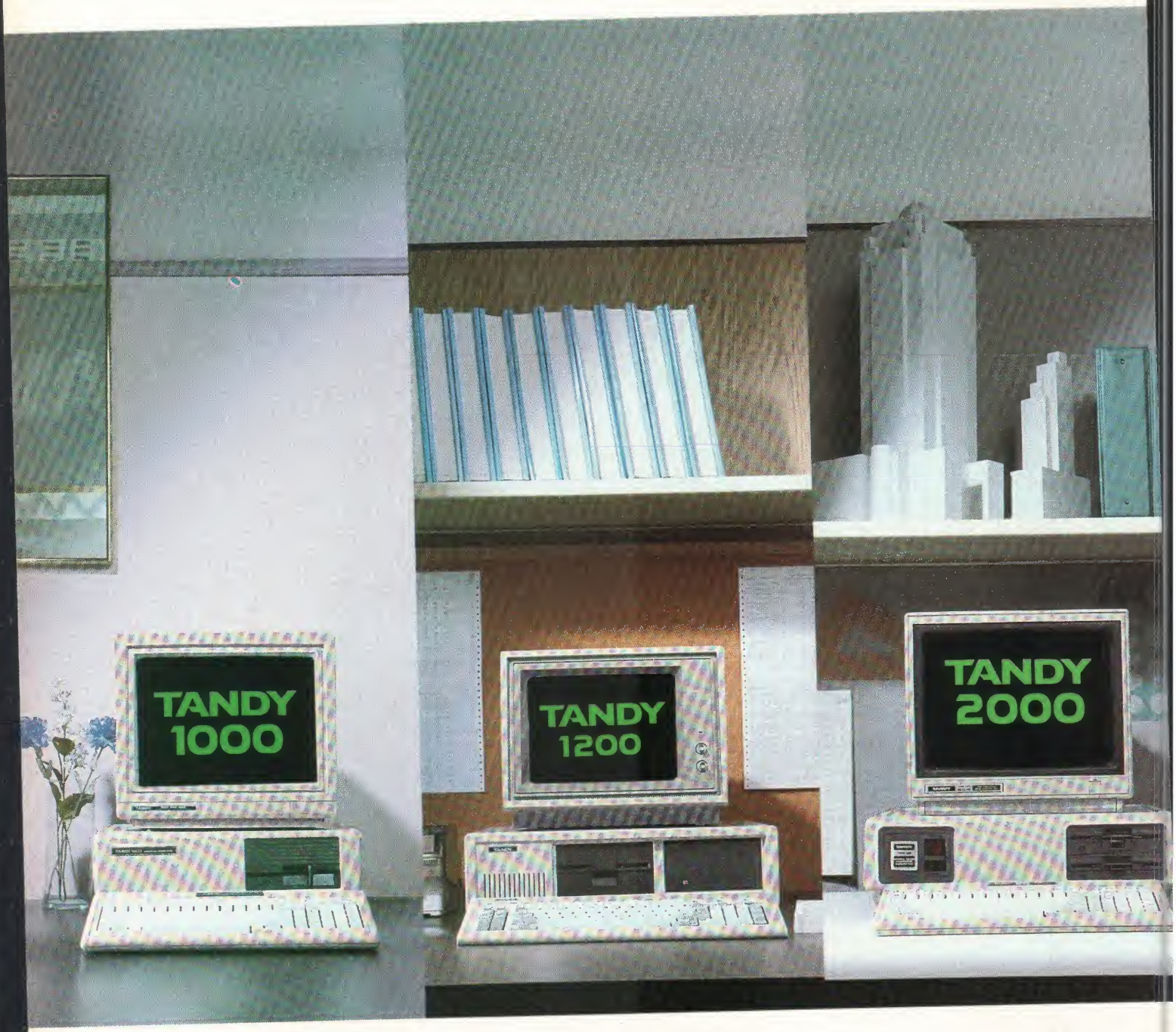
Tandy is a registered trademark of Tandy Corp.

American Cryptronics, Inc.

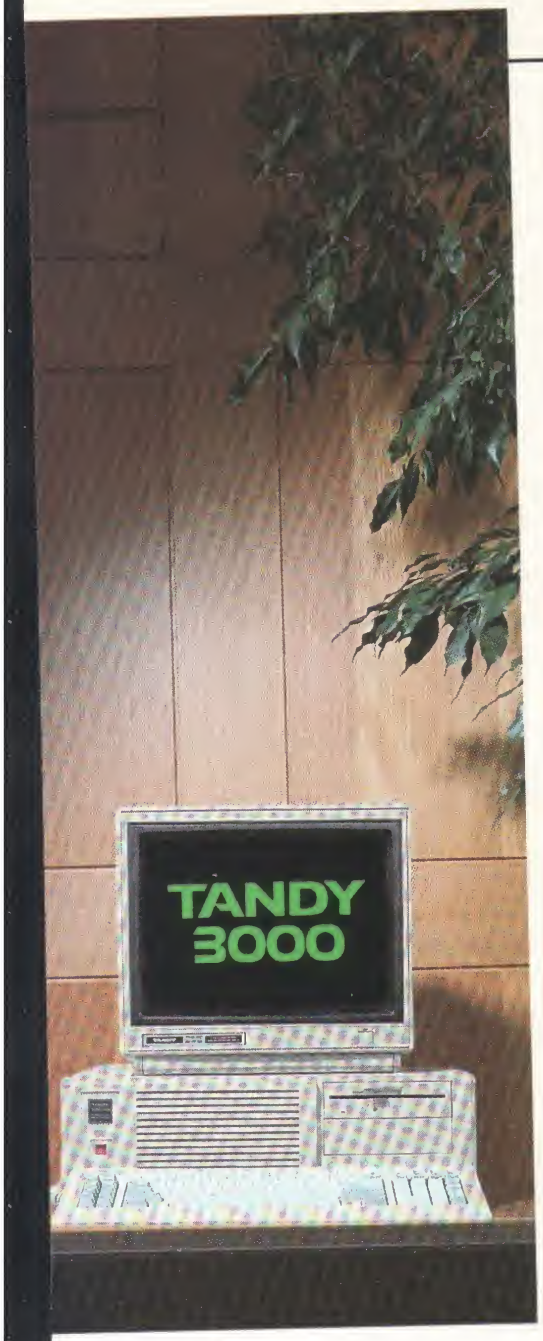
(Formerly Cryptronics, Inc.)
1580 Corporate Drive, Suite 123
Costa Mesa, California 92626
(714) 540-1174

```
682 IF Q$="Y" OR Q$="y" THEN 684 ELSE 412
684 COLOR C(1):LOCATE 15,1Ø:PRINT"Send Data to Printer (Y/N)";:COLOR C(2):INPUT
PT$
686 COLOR C(1):LOCATE 17,1Ø:PRINT"What Month (Ø1-12): ";:COLOR C(2):INPUT M1$:M1
=VAL(M1$)
688 CLS:LOCATE 1,31:COLOR C(1):PRINT"Year-to-Date Summary":LOCATE 1,6Ø:PRINT DAT
E$:Z=Ø:Y=3
69Ø IF PT$="Y" OR PT$="y" THEN GOSUB 726
692 GOSUB 936
694 LOCATE 3,29:PRINT"INCOME: ":COLOR C(2)
696 IF EOF(1)=-1 THEN CLOSE #1:GOTO 712
698 Z=Z+1:INPUT#1,IT$(Z),BAL(Z):TT=TT+BAL(Z)
7ØØ IF IT$(Z)=EX$(1) THEN Y=Y+1:TT=TT-BAL(Z):LOCATE Y,29:COLOR C(1):PRINT"TOTAL
INCOME: ":LOCATE Y,43:COLOR C(2):PRINT USING K$;TT:TT=Ø:TT=BAL(Z)
7Ø2 IF IT$(Z)=EX$(1) THEN Y=Y+2:COLOR C(1):LOCATE Y,29:PRINT"EXPENSES: ":COLOR C
(2):IF PT$="QX" THEN LPRINT TAB(51) "_____":LPRINT TAB(34) "TOTAL INCOME:
 ";:LPRINT USING K$;TT:LPRINT "":LPRINT TAB(3Ø) "EXPENSES: ":TT=Ø
7Ø4 Y=Y+1:COLOR C(2):LOCATE Y,29:PRINT IT$(Z):LOCATE Y,43:PRINT USING K$;BAL(Z)
7Ø6 IF Y>22 THEN GOSUB 736:Y=2:CLS:COLOR C(1):LOCATE 1,3Ø:PRINT"Year-to-Date Sum
mary":LOCATE 1,6Ø:PRINT DATE$
7Ø8 IF PT$="QX" THEN GOSUB 728
71Ø TZ=TZ+BAL(Z):GOTO 696
712 Y=Y+1:LOCATE Y,29:COLOR C(1):PRINT"TOTAL EXPENSES: ":COLOR C(2):LOCATE Y,43:
PRINT USING K$;TT:IF PT$="QX" THEN LPRINT TAB(51) "_____":LPRINT TAB(34) "T
OTAL EXPENSES: ";:LPRINT USING K$;TT:LPRINT ""
714 Y=Y+2:IF Y>22 THEN GOSUB 728:Y=2:CLS
716 LOCATE Y,3Ø:COLOR C(1):PRINT"Balance:     ";:COLOR C(2):PRINT USING K$;TZ
718 IF Y>22 THEN GOSUB 736:Y=2
72Ø IF PT$="QX" THEN 734
```

```
722 GOSUB 900
724 TZ=0:GOTO 412
726 GOSUB 666:LPRINT TAB(30) "Year to Date Summmary - ";MTH$(M1):LPRINT TAB(41)
DATE$:GOSUB 666:PT$="QX":LPRINT TAB(30) "INCOME: ":RETURN
728 LPRINT TAB(30);IT$(Z);"          ";
730 L1=LEN(IT$(Z)):P1=11-L1:FOR V=0 TO P1:LPRINT " ";:NEXT V
732 LPRINT USING K$;BAL(Z):RETURN
734 GOSUB 666:LPRINT TAB(51) "==========":LPRINT TAB(34) "BALANCE:          ";:LPR
INT USING K$;TZ:GOSUB 666:GOSUB 900:GOTO 412
736 LOCATE 24,30:COLOR C(1):PRINT"Hit <ENTER> to Continue";:COLOR C(2):GOSUB 900
:RETURN
738 '
740 '*************************************************************************
742 ' *** PROPERTY INFORMATION ***
744 '*************************************************************************
746 '
748 CLS:COLOR C(1):LOCATE 12,30:PRINT"Rental Property Information":COLOR C(2)
750 TAG$="" :LOCATE 14,30:PRINT"1) Input Data":LOCATE 15,30:PRINT"2) View Data"
752 LOCATE 16,30:PRINT"3) Change Data":LOCATE 17,30:PRINT"4) END Job"
754 LOCATE 19,30:PRINT"(1, 2, 3, or 4)":GOSUB 900
756 ON VAL(AT$) GOSUB 758,832,880,904:GOTO 748
758 'SUBROUTINE TO INPUT DATA ON DISK
760 GOSUB 892
762 REC=LOF(1)/200
764 REC=REC+1
766 CLS:COLOR C(1):LOCATE 1,30:PRINT"Rental Property Information"
768 LOCATE 3,10:PRINT"Property Code No: ":LOCATE 3,55:PRINT"Update: ":LOCATE 4,1
9:PRINT"Address: ":LOCATE 4,55:PRINT"Record #":LOCATE 5,22:PRINT"City: ":LOCATE
6,21:PRINT"State: "
770 LOCATE 7,18:PRINT"Zip Code: ":LOCATE 9,13:PRINT"Renter's Name: ":LOCATE 10,1
5:PRINT"Rental Date: ":LOCATE 11,10:PRINT"Phone (XXX-XXXX): "
772 LOCATE 12,14:PRINT"Deposit Paid: ":LOCATE 13,14:PRINT"Monthly Rent: ":LOCATE
 15,13:PRINT"Purchase Date: ":LOCATE 16,12:PRINT"Purchase Price: "
774 LOCATE 18,1:PRINT"Lien #1: ":LOCATE 18,30:PRINT"Date Due (XX): ":LOCATE 18,5
0:PRINT"Monthly Mortgage: "
776 LOCATE 19,1:PRINT"Lien #2: ":LOCATE 19,30:PRINT"Date Due (XX): ":LOCATE 19,5
0:PRINT"Monthly Mortgage: ":COLOR C(2)
778 IF TAG$="VIEW" OR TAG$="CHANGE" THEN 846
780 P$=COD$(REC):A$=ADR$(REC):N$=NAM$(REC)
782 LOCATE 3,30:PRINT P$:LOCATE 3,63:PRINT DATE$:LOCATE 4,30:PRINT A$:LOCATE 4,6
3:PRINT REC:LOCATE 9,30:PRINT N$
784 LOCATE 5,28:INPUT C$:IF C$="" THEN C$=CITY$
786 LOCATE 6,28:INPUT S$:IF S$="" THEN S$=STAT$
788 LOCATE 7,28:INPUT Z$:IF Z$="" THEN Z$=ZIP$
790 LOCATE 10,28:INPUT R$:IF R$="" THEN R$=RDAT$
792 LOCATE 11,28:INPUT H$:IF H$="" THEN H$=PHON$
794 LOCATE 12,28:INPUT D(REC):IF D(REC)=0 THEN D(REC)=CVD(DEP$)
796 LOCATE 13,28:INPUT M(REC):IF M(REC)=0 THEN M(REC)=CVD(MREN$)
798 LOCATE 15,28:INPUT E$:IF E$="" THEN E$=PUR$
800 LOCATE 16,28:INPUT PRI(REC):IF PRI(REC)=0 THEN PRI(REC)=CVD(PRIC$)
802 LOCATE 18,10:INPUT LN$:IF LN$="" THEN LN$=LIEN$
804 LOCATE 18,45:INPUT PY$:IF PY$="" THEN PY$=PAY$
806 LOCATE 18,67:INPUT Q(REC):IF Q(REC)=0 THEN Q(REC)=CVD(MORT$)
808 LOCATE 19,10:INPUT LM$(REC):IF LM$(REC)="" THEN LM$(REC)=LF$
810 LOCATE 19,45:INPUT PZ$(REC):IF PZ$(REC)="" THEN PZ$(REC)=PB$
812 LOCATE 19,67:INPUT Q1(REC):IF Q1(REC)=0 THEN Q1(REC)=CVD(MP$)
814 LSET UDAT$=DATE$:LSET COD$=P$:LSET ADR$=A$:LSET CITY$=C$:LSET STAT$=S$:LSET
ZIP$=Z$:LSET NAM$=N$:LSET RDAT$=R$:LSET PHON$=H$:LSET DEP$=MKD$(D(REC))
816 LSET MREN$=MKD$(M(REC)):LSET PUR$=E$:LSET PRIC$=MKD$(PRI(REC))
```



We edged out the competition with better solutions for every application.



Presenting the Tandy® line of PC-compatible computers from Radio Shack

At Radio Shack Computer Centers, you won't find just any computer, you'll find the *right* computer for your business. Our line of computers based on the MS™-DOS operating system lets you outfit your office with the computing power you need, without paying for capabilities you may never use. And since we carry a wide range of software and accessories, you can do all of your business at one location. So come in today and let our trained staff help you decide which Tandy computer is right for your business.

America's #1 PC compatible

Looking for a complete, low-cost MS-DOS system? Take a look at our popular Tandy 1000. Because the Tandy 1000 is compatible with the IBM® PC, there are thousands of software programs for you to choose from. But unlike IBM's PC, every Tandy 1000 comes with DeskMate® software. DeskMate includes six applications so you can start computing from day one.

The PC/XT's "mirror image"

The Tandy 1200 works like an IBM PC/XT—but costs much less. The 1200 runs the same top-name software and supports the same hardware as the PC/XT. So if you're already using one or more IBM PC's in your office, the Tandy 1200 lets you expand for less—and continue using the same software.

The graphics advantage

Searching for a quality computer with advanced graphics capabilities? Take a good look at the affordable Tandy 2000. The 2000 features high-resolution graphics to create razor-sharp graphs and diagrams in a brilliant array of colors.

Combined with computer-aided-design software, a Tandy 2000 gives your company endless opportunities for increased creativity and productivity.

The new Tandy 3000

Introducing a super-powerful computer that's the affordable alternative to the IBM PC/AT. The Tandy 3000 has all the power you need to manage your business, and it's compatible with programs designed for the PC/AT and PC/XT. The Tandy 3000 operates at 8 megahertz, twice the speed of the industry standard. It comes with 512K of main memory and ten expansion slots. And with the forthcoming XENIX™ operating system, two to six users will be able to use the Tandy 3000 simultaneously, using low-cost data terminals.

In business...for business

For the best in state-of-the-art computers, popular software, professional training, support and affordable leasing, visit your local Radio Shack Computer Center today.

Available at over 1200 Radio Shack Computer Centers and at participating Radio Shack stores and dealers.

Radio Shack®
COMPUTER CENTERS
A DIVISION OF TANDY CORPORATION

New! 1986 Computer Catalog.
Mail To: Radio Shack, Dept. 86-A-139
300 One Tandy Center, Fort Worth, TX 76102

Name _____
Company _____
Address _____
City _____
State _____ ZIP _____
Phone _____

```
818 LSET LIEN$=LN$:LSET PAY$=PY$:LSET MORT$=MKD$(Q(REC))
82Ø LSET LF$=LM$(REC):LSET PB$=PZ$(REC):LSET MP$=MKD$(Q1(REC))
822 PUT #1,REC
824 IF TAG$="CHANGE" THEN GOTO 858
826 COLOR C(1):LOCATE 21,3Ø:PRINT"1) Input Another File"
828 LOCATE 22,3Ø:PRINT"2) Change Information":LOCATE 23,3Ø:PRINT"3) END Job":LOC
ATE 24,3Ø:PRINT"(1, 2, or 3)";:GOSUB 9ØØ
83Ø ON VAL(AT$) GOSUB 764,878,888:GOTO 826
832 'SUBROUTINE FOR VIEWING INFORMATION FILES
834 GOSUB 892:TAG$="VIEW" :REC=1:Y=1Ø:T=Ø:GOSUB 9Ø6
836 LOCATE 8,25:PRINT"What is the Record No. (XX/Ø=All)";:COLOR C(2):INPUT MN$
838 REC=VAL(MN$):IF REC=Ø THEN REC=1
84Ø TAG$="VIEW":GET #1,REC:IF EOF(1)=-1 THEN CLOSE#1:GOTO 748
842 PRI(REC)=CVD(PRIC$):M(REC)=CVD(MREN$):D(REC)=CVD(DEP$):Q(REC)=CVD(MORT$):Q1(
REC)=CVD(MP$)
844 GOTO 766
846 COLOR C(2):LOCATE 3,3Ø:PRINT COD$:LOCATE 4,3Ø:PRINT ADR$:LOCATE 5,3Ø:PRINT C
ITY$:LOCATE 6,3Ø:PRINT STAT$:LOCATE 7,3Ø:PRINT ZIP$:LOCATE 9,3Ø:PRINT NAM$
848 LOCATE 3,63:PRINT UDAT$:LOCATE 4,63:PRINT REC
85Ø LOCATE 1Ø,3Ø:PRINT RDAT$:LOCATE 11,3Ø:PRINT PHON$:LOCATE 12,3Ø:PRINT USING K
K$;D(REC):LOCATE 13,3Ø:PRINT USING KK$;M(REC):LOCATE 15,3Ø:PRINT PUR$:LOCATE 16,
3Ø:PRINT USING K$;PRI(REC)
852 LOCATE 18,1Ø:PRINT LIEN$:LOCATE 18,45:PRINT PAY$:LOCATE 18,67:PRINT USING KK
$;Q(REC)
854 LOCATE 19,1Ø:PRINT LF$:LOCATE 19,45:PRINT PB$:LOCATE 19,67:PRINT USING KK$;Q
1(REC)
856 IF TAG$="CHANGE" THEN 78Ø
858 LOCATE 24,22:COLOR C(1):PRINT"<N>ext, <P>revious, <M>enu or <C>hange";
86Ø COLOR C(2):AN$=INKEY$:IF AN$="" THEN 86Ø
862 IF AN$="N" OR AN$="n" THEN 87Ø
864 IF AN$="p" OR AN$="P" THEN REC=REC-2:GOTO 87Ø
866 IF AN$="C" OR AN$="c" THEN 89Ø
868 IF AN$="M" OR AN$="m" THEN 876 ELSE 86Ø
87Ø REC=REC+1:IF EOF(1)=-1 THEN CLOSE#1:GOTO 748
872 IF REC<Ø THEN CLOSE #1:GOTØ 712Ø
874 GOTO 84Ø
876 CLOSE#1:GOTO 748
878 'SUBROUTINE TO CHANGE DATA
88Ø CLS:LOCATE 6,24:COLOR C(1):PRINT"Change Property Information":GOSUB 892:TAG$
="CHANGE":REC=1:Y=1Ø:T=Ø:GOSUB 9Ø8
882 LOCATE 8,25:PRINT"What is the Record No. ";:COLOR C(2)
884 INPUT MN$:REC=VAL(MN$):IF REC<Ø THEN 882
886 GOTO 84Ø
888 CLOSE #1:GOTO 748
89Ø TAG$="CHANGE":GOTO 766
892 OPEN"R",#1,"A:RENTAL.DAT",2ØØ
894 FIELD#1,3 AS COD$,1Ø AS UDAT$,25 AS ADR$,15 AS CITY$,2 AS STAT$,5 AS ZIP$,2Ø
 AS NAM$,8 AS PHON$,8 AS RDAT$,1Ø AS MREN$,1Ø AS DEP$,1Ø AS PUR$,1Ø AS PRIC$,1Ø
AS MORT$,2Ø AS LIEN$,2 AS PAY$,2Ø AS LF$,2 AS PB$,1Ø AS MP$
896 RETURN:CLOSE #1:GOTO 748
898 OPEN "R",#1,"A:JOURN"+M1$+".FIL",7Ø:FIELD #1,3 AS COD$,1Ø AS CAT$,5 AS DT$,4
 AS DL$,17 AS PDTO$,2Ø AS DSC$,11 AS AMT$:RETURN
9ØØ AT$=INKEY$:IF AT$="" THEN 9ØØ
9Ø2 RETURN
9Ø4 GOTO 26
9Ø6 CLS:LOCATE 6,2Ø:COLOR C(1):PRINT"Do You Want to See All Files or One Record
No. ? "
9Ø8 IF COD$(REC)="" THEN REC=1:RETURN
91Ø Y=Y+1:COLOR C(1):LOCATE Y,T+1:PRINT USING K2$;REC:LOCATE Y,T+5:PRINT COD$(RE
```

```
C):LOCATE Y,T+1Ø:PRINT ADR$(REC)
912 REC=REC+1:IF Y>21 THEN Y=1Ø:T=4Ø:GOTO 9Ø8
914 GOTO 9Ø8
916 COLOR C(1):LOCATE 1,6Ø:PRINT"##  ACCOUNT":L=2 : Y=1
918 LOCATE L,6Ø:PRINT USING K2$;Y;:LOCATE L,64:PRINT EX$(Y):L=L+1 : Y=Y+1
92Ø IF EX$(Y)= "" THEN 924
922 GOTO 918
924 L=Ø:COLOR C(2):RETURN
926 LOCATE 1,61:COLOR C(1):PRINT"##  ACCOUNT":L=2:Y=1
928 LOCATE L,61:PRINT USING K2$;Y;:LOCATE L,65:PRINT IN$(Y):L=L+1:Y=Y+1
93Ø IF IN$(Y)="" THEN 934
932 GOTO 928
934 L=Ø:COLOR C(2):RETURN
936 OPEN"I",#1,"A:SUMMARY.FIL":RETURN
938 OPEN"O",#1,"A:SUMMARY.FIL":RETURN
94Ø ON ERROR GOTO 1Ø1Ø
942 X=Ø:OPEN"I",#1,"A:PROPERTY.ACC"
944 IF EOF(1)=-1 THEN CLOSE#1:GOTO 948
946 X=X+1:INPUT#1,COD$(X),ADR$(X),NAM$(X):GOTO 944
948 X=Ø:OPEN"I",#1,"A:INCOME.ACC"
95Ø IF EOF(1)=-1 THEN CLOSE#1:GOTO 954
952 X=X+1:INPUT#1,IN$(X):GOTO 95Ø
954 X=Ø:OPEN"I",#1,"A:EXPENSE.ACC"
956 IF EOF(1)=-1 THEN CLOSE#1:GOTO 26
958 X=X+1:INPUT#1,EX$(X):GOTO 956:CLOSE#1:RETURN
96Ø CLS:COLOR C(3):IF ERR<>53 THEN 964:LOCATE 5,17:PRINT"Files NOT Found!! - Do
One of the Following:":LOCATE 9,1Ø:PRINT"<BREAK> Program Execution and Insert Fi
le Disk in Drive A"
962 LOCATE 11,36:PRINT"or":LOCATE 13,15:PRINT"Select Main Menu #4 to Set-up Acco
unts!!"
964 GOSUB 9ØØ:CLOSE:COLOR C(2):GOTO 26
966 OPEN"O",#1,"A:PROPERTY.ACC":RETURN
968 OPEN"I",#1,"A:PROPERTY.ACC":RETURN
97Ø OPEN"O",#2,"A:TEMP.ACC":RETURN
972 KILL"A:PROPERTY.ACC"
974 NAME "A:TEMP.ACC" AS "A:PROPERTY.ACC":GOTO 68
976 OPEN"O",#1,"A:EXPENSE.ACC":RETURN
978 OPEN"I",#1,"A:EXPENSE.ACC":RETURN
98Ø KILL"A:EXPENSE.ACC"
982 NAME"A:TEMP.ACC"AS"A:EXPENSE.ACC":GOTO 56
984 OPEN"O",#1,"A:INCOME.ACC":RETURN
986 OPEN"I",#1,"A:INCOME.ACC":RETURN
988 KILL"A:INCOME.ACC"
99Ø NAME"A:TEMP.ACC"AS"A:INCOME.ACC":GOTO 56
992 CLS:LOCATE 5,15:PRINT"PREVIOUSLY ENTERED PROPERTY CODES WILL BE LOST!!"
994 LOCATE 7,25:PRINT"Do You Want to Continue (Y/N)":COLOR C(2)
996 AZ$=INKEY$ : IF AZ$="" THEN 996
998 IF AZ$= "N" OR AZ$= "n" THEN 26
1ØØØ IF AZ$= "Y" OR AZ$= "y" THEN RETURN ELSE 26
1ØØ2 RETURN
1Ø1Ø IF ERR=71 THEN COLOR C(3):PRINT"PLACE DATA DISKETTE IN DRIVE A":COLOR C(2):
GOTO 1Ø2Ø
1Ø11 IF ERR=27 THEN COLOR C(3):PRINT"PRINTER IS OUT OF PAPER - RESTOCK":COLOR C(
2):GOTO 1Ø2Ø
1Ø12 IF ERR=61 THEN COLOR C(3):PRINT"DISK IS FULL-INSERT ANOTHER FORMATTED DISK"
:COLOR C(2):GOTO 1Ø2Ø
1Ø13 IF ERR=7Ø THEN COLOR C(3);PRINT"WRITE PROTECT NOTCH IS COVERED - REMOVE DIS
K AND REMOVE!":COLOR C(2):GOTO 1Ø2Ø
1Ø14 IF ERR=53 THEN COLOR C(3):PRINT"IF YOU ARE FIRST TIME USER, YOU WILL BE GOI
NG TO -Set up Accounts-                IF THIS NOT CORRECT, PLACE CORRECT FILE DIS
K IN DRIVE A AND RESTART!":COLOR C(2):FOR JJ=1 TO 1ØØØØ:NEXT JJ:GOTO 5Ø
1Ø19 COLOR C(3):PRINT "ERROR DETECTED - TRY AGAIN!":COLOR C(2):GOTO 26
1Ø2Ø LOCATE 24,3Ø:COLOR C(1):PRINT"Hit <ENTER> to Continue";:COLOR C(2):GOSUB 9Ø
Ø:RESUME
```

Three BASIC programs for the time of your life

Your 100 Takes a Licking and Keeps on Ticking

By Nathaniel F. Ireland

The Model 100's TIME$ function indicates the passage of time in one-second intervals. For those interested in smaller intervals, the BASIC program *TIMER.BA* (Listing 1) indicates elapsed time in hundredth-second intervals. Its accuracy can be adjusted within 0.1 second (+ or - 0.28 percent) over the program's three-minute maximum range. This range can be extended but the change produces an increased error percentage, which is directly proportional to the increase in range. An increase in range to six minutes may produce an inaccuracy of + or - 0.55 percent.

The Program

The heart of the program is the precisely tuned timing loop, lines 200 through 230. The timing loop without lines 210 and 220 runs much too fast; the number of pound signs within these lines fine-tune the loop.

The balance of the program is straightforward and the operation of the various sections is described in REM statements. These REM statements may be removed from the program to conserve memory.

Fine Tuning the Timing Loop

Run the calibrate feature listed on the menu by pressing 'T'. Do this several times. If the number of counts per second is consistently less than 100, slowly decrease the number of pound signs in lines 210 and 220 until the count just equals 100. Run the calibrate feature several times. If the count per second consistently equals 100, the loop is fine-tuned. If the counts do not equal 100, decrease the number of pound signs very slowly until they do.

If the count is 100 or greater, increase the number of pound signs until the count drops below 100, then proceed as above.

Increasing the Program's Range

The maximum range of the program is determined by the right-most value of NN in Line 130. For each increase of 6,000 in the value of NN, the program's range is increased by one minute. Don't forget to increase the value in Line 410 by a like amount.

Have you wondered if your vehicle's speedometer is accurate? *STIMER.BA* (Listing 2) and your Model 100 can answer this question. There are two more requisites: a road with a measured mile and someone to assist you. If you don't know of a measured mile near you, your state police can provide this information. Some states have mile posts that can also be used. For safety's sake, the other person should either drive the vehicle or operate the Model 100.

Using the Program

After loading *STIMER.BA*, check its calibration using the 'T' function. While driving, bring your vehicle up to about 40 mph before approaching the start of the measured mile. As your vehicle passes the starting point, tap the Model 100 space bar. Carefully hold the vehicle's speed constant and press the space bar again as the vehicle passes the end of the measured mile. Your Model 100 automatically writes the calculated miles-per-hour rate to RAM using the name RUN 1.DO and resets itself for a run at another speed through the measured mile. Write down or remember your speedometer reading of the first run.

Make another run through the measured mile at another speed, say 55 mph, following the same procedure. This time the Model 100 stores the data in RAM as RUN 2.DO. Successive runs can be made in a like manner. The text files in RAM can later be compared with your speedometer readings to yield an accurate calibration of your vehicle's speedometer.

Designed for athletic events in which the contestants compete against time, *ETIMER.BA* (Listing 3) keeps their individual times for comparison at the end of the event

Nathaniel Ireland, now a retired gentleman farmer, was an engineer in the electronics industry for many years. In addition to his agricultural hobby, he finds time to enjoy his computers and do some consulting work.

to determine placing. Examples of such events would be downhill and slalom in skiing and barrel racing in rodeo.

Program Operation

When *ETIMER. BA* is run, it first asks for the event title. In a maximum of five letters, enter the event name. If the RAM storage option is selected later, this name with the .DO extension will be the name of the test file for the event.

The second question asked is the contestant's identifier (use up to a maximum of 10 characters). Once this is entered, the timing mode for the contestant is enabled. Tap the space bar to start the timer, then tap the space bar again to stop the timer. The Model 100 records this data into an array and returns to the second question for another contestant's identifier.

When all the contestants for the event have been timed, enter END to the second question. The program sorts the array in ascending order according to their times and creates a new array. The program prints the new array on the screen, indicating the contestant's identifier, the respective time and place in the event.

A question to record the array data to RAM is then asked. An 'N' (no) answer returns the program to the first question. An END response returns you to the Model 100's menu. A new event title restarts the program.

A 'Y' (yes) answer to this question records the array data to RAM using the event title in the first question with the .DO extension, then the program returns to the first question.

The data file recorded to RAM can be viewed and sent to a printer in a 40-column format (use SHIFT PRINT). In addition to the information seen on the screen display, the date and time of day are recorded and sent to the printer.□

Listing 1: PCM BAR CODED LISTING

```
1Ø CLS
2Ø PRINT@1Ø,"* INTERVAL  TIMER *"
3Ø PRINT:PRINTTAB(4)"Tap space bar to st
art TIMER."
5Ø PRINTTAB(4)"Tap space bar again to st
op TIMER."
6Ø PRINT:PRINT" or Tap <T> key to check
TIMER cali-          bration. Wait 3Ø sec
onds."
7Ø ' Written by N. F. Ireland, January 2
4, 1985
1ØØ REM START TIMER, TIMER LOOP & STOP
       TIMER
11Ø A$=TIME$
12Ø X$=INKEY$:IFX$=""THEN11Ø
13Ø IFX$="T"THENNN=3ØØØELSENN=18ØØØ
2ØØ FORN=1TONN
21Ø C$="###############"
22Ø D$="#########"
23Ø Y$=INKEY$:IFY$=""THENNEXT
24Ø B$=TIME$
3ØØ REM COMPUTE PULSE COUNT/SECOND,
       DISPLAY & END
31Ø IFX$="T"THEN32ØELSE41Ø
32Ø N=N-1
33Ø T1=VAL(RIGHT$(B$,2)):T2=VAL(RIGHT$(A
$,2)):IFT1<T2THENT1=T1+6Ø
35Ø ET=T1-T2
36Ø CO=N/ET
37Ø CLS:PRINT@85,"COUNTS = "CO" PER SECO
ND.":PRINT:PRINT"THIS COUNT MUST BE 1ØØ.
 IF NOT, INTERNALPROGRAM ADJUSTMENT MAY
BE REQUIRED. TRY":PRINT"IT AGAIN. PRESS
<F4> TO RE-RUN TIMER."
38Ø END
4ØØ REM OUT OF TIMER RANGE DISPLAY &
       END
41Ø IFN>18ØØØTHENCLS:PRINT@24Ø,"ELAPSED
TIME OUT OF RANGE OF TIMER.":BEEP:END
5ØØ REM COMPUTE & DISPLAY ELAPSED TIME
51Ø N=N-1:SE=N/1ØØ:IFSE>59THENMI=FIX(SE/
6Ø):SE=SE-(6Ø*MI)
52Ø CLS:PRINT@12Ø,"ELAPSED TIME ="MI"MIN
UTE,"SE"SECONDS.":PRINT:PRINT:PRINT"PRES
S <F4> TO RE-RUN TIMER.":END
```

Listing 2: PCM BAR CODED LISTING

```
5 MAXFILES=1:NT=1
1Ø CLS
2Ø PRINT@4,"* SPEEDOMETER CHECK  TIMER *
"
3Ø PRINT:PRINTTAB(4)"Tap space bar to st
art TIMER."
5Ø PRINTTAB(4)"Tap space bar again to st
op TIMER."
6Ø PRINT:PRINT" or Tap <T> key to check
TIMER cali-          bration. Wait 3Ø sec
onds."
7Ø ' Written by N. F. Ireland, January 2
4, 1985
1ØØ REM START TIMER, TIMER LOOP & STOP
       TIMER
11Ø A$=TIME$
12Ø X$=INKEY$:IFX$=""THEN11Ø
13Ø IFX$="T"THENNN=3ØØØELSENN=18ØØØ
2ØØ FORN=1TONN
21Ø C$="###############"
22Ø D$="#########"
23Ø Y$=INKEY$:IFY$=""THENNEXT
24Ø B$=TIME$
3ØØ REM COMPUTE PULSE COUNT/SECOND,
       DISPLAY & END
31Ø IFX$="T"THEN32ØELSE41Ø
32Ø N=N-1
33Ø T1=VAL(RIGHT$(B$,2)):T2=VAL(RIGHT$(A
$,2)):IFT1<T2THENT1=T1+6Ø
35Ø ET=T1-T2
36Ø CO=N/ET
37Ø CLS:PRINT@85,"COUNTS = "CO" PER SECO
ND.":PRINT:PRINT"THIS COUNT MUST BE 1ØØ.
 IF NOT, INTERNALPROGRAM ADJUSTMENT MAY
```

```
BE REQUIRED. TRY":PRINT"IT AGAIN. PRESS
<F4> TO RE-RUN TIMER."
38Ø END
4ØØ REM OUT OF TIMER RANGE DISPLAY &
      END
41Ø IFN>18ØØØTHENCLS:PRINT@24Ø,"ELAPSED
TIME OUT OF RANGE OF TIMER.":BEEP:END
5ØØ REM COMPUTE ELAPSED TIME
51Ø N=N-1:SE=N/1ØØ:IFSE>59THENMI=FIX(SE/
6Ø):SE=SE-(6Ø*MI)
6ØØ REM COMPUTE MPH, RECORD DATA TO RAM
AND RE-RUN PROGRAM
61Ø SC=N/1ØØ:MPH=1/(SC/36ØØ)
62Ø OPEN"RAM:MPH"+STR$(NT)+".DO"FOROUTPU
TAS1
63Ø PRINT#1,"   TIME = "MI" MINUTES,";SE
 "SECONDS"
64Ø A$="     MILES PER HOUR =":PRINT#1,U
SING"\                    \ ####.##";A
$,MPH
65Ø CLOSE:NT=NT+1:GOTO1Ø
```

Listing 3:

```
5 MAXFILES=1:DIMCI$(2Ø),CE$(2Ø),MI(2Ø),M
E(2Ø),DO(2Ø),SE(2Ø),SC(2Ø),SD(2Ø)
1Ø CLS
15 CLS:PRINT@13,"EVENT TIMER":PRINT@12Ø,
;:INPUT"ENTER EVENT TITLE (5 CHRS MAX)";
ET$:CLS:IFLEN(ET$)>5THEN15ELSEIFET$="END
"THENMAXFILES=Ø:MENU
2Ø PRINT@13,"* EVENT  TIMER *"
25 PRINT:INPUT"ENTER CONTESTENT IDENTIFI
ER (1Ø CHRS MAX) ";CI$:CLS:PRINT@15,CI$:
IFCI$="END"THEN7ØØ
3Ø PRINTTAB(4)"Tap space bar to start TI
MER."
5Ø PRINTTAB(4)"Tap space bar again to st
op TIMER."
6Ø PRINT:PRINT" or Tap <T> key to check
TIMER cali-          bration. Wait 3Ø sec
onds."
7Ø ' Written by N. F. Ireland, January 2
4, 1985
1ØØ REM START TIMER, TIMER LOOP & STOP T
IMER
11Ø A$=TIME$
12Ø X$=INKEY$:IFX$=""THEN11Ø
13Ø IFX$="T"THENNN=3ØØØELSENN=18ØØØ
2ØØ FORN=1TONN
21Ø C$="###############"
22Ø D$="#########"
23Ø Y$=INKEY$:IFY$=""THENNEXT
24Ø B$=TIME$
3ØØ REM COMPUTE PULSE COUNT/SECOND, DISP
LAY & END
31Ø IFX$="T"THEN32ØELSE41Ø
32Ø N=N-1
33Ø T1=VAL(RIGHT$(B$,2)):T2=VAL(RIGHT$(A
$,2)):IFT1<T2THENT1=T1+6Ø
35Ø ET=T1-T2
36Ø CO=N/ET
37Ø CLS:PRINT@85,"COUNTS = "CO" PER SECO
ND.":PRINT:PRINT"THIS COUNT MUST BE 1ØØ.
 IF NOT, INTERNALPROGRAM ADJUSTMENT MAY
BE REQUIRED. TRY":PRINT"IT AGAIN. PRESS
<F4> TO RE-RUN TIMER."
38Ø END
4ØØ REM OUT OF TIMER RANGE DISPLAY, BEEP
 & RE-RUN
41Ø IFN>18ØØØTHENCLS:PRINT@24Ø,"ELAPSED
TIME OUT OF RANGE OF TIMER.":FORD=ØTO2Ø:
BEEP:NEXTD:RUN
5ØØ REM COMPUTE & DISPLAY ELAPSED TIME
51Ø N=N-1:SE=N/1ØØ:SC=SE:IFSE>59THENMI=F
IX(SE/6Ø):SC=SE-(6Ø*MI)
52Ø CLS:PRINT@12Ø,"ELAPSED TIME ="MI"MIN
UTE,"SC"SECONDS.":FORD=ØTO1ØØØ:NEXT
6ØØ REM STORE CONTESTANT INFORMATION
61Ø J=J+1:CI$(J)=CI$:MI(J)=MI:SC(J)=SC:S
E(J)=SE
67Ø IFJ=NCTHEN7ØØELSECLS:GOTO25
7ØØ REM SORT ARRAY
7Ø5 Q=1
71Ø FORSJ=1TOJ
72Ø  FORIJ=1TOJ
73Ø  IFSE(SJ)>SE(IJ)THENQ=Q+1
74Ø  NEXTIJ
75Ø DO(Q)=Q:CE$(Q)=CI$(SJ):ME(Q)=MI(SJ):
SD(Q)=SC(SJ):Q=1
76Ø NEXTSJ
8ØØ REM PRINT SORTED ARRAY
81Ø CLS
82Ø FORNN=1TOJ
83Ø PRINTUSING"  ##   \          \   #,
   ##,.##";DO(NN),CE$(NN),ME(NN),SD(NN):
CC=CC+1
84Ø IFCC=7THENINPUT"PRESS ENTER TO CONTI
NUE";CC$:CLS:CC=Ø
85Ø NEXTNN
86Ø PRINT@12Ø,;:INPUT"RECORD TO RAM (Y/N
)";OP$:IFOP$="Y"THEN9ØØELSEIFOP$="N"THEN
RUNELSE86Ø
9ØØ REM RECORD DATA TO RAM
91Ø OPEN"RAM:"+ET$+".DO"FOROUTPUTAS1
915 PRINT#1,"              ";ET$" EVENT"
917 PRINT#1,"  ";DATE$;"
    ";TIME$
92Ø PRINT#1,USING"\      \\        \\
      \\     \";" PLACE","CONTESTANT"
,"       MIN","  SEC"
93Ø FORSR=1TOJ
94Ø PRINT#1,USING"  #,     \          \
    #,   ##,.##";DO(SR),CE$(SR),ME(SR),
SD(SR)
95Ø NEXTSR
96Ø CLOSE:RUN
```

PCM

Make melodies to soothe the savage hacker . . .

The Music Machine

By Jerry Stajduhar

The *Music Machine* allows the user to create, modify and save songs using a symbolic notation and a full screen editor. After the song has been completed, *Music Machine* draws the notes, and if the user has access to a Radio Shack CGP-220 printer, the music sheet can be printed (see Figure 1). The program was written in BASIC on a single drive Tandy 1000 with 256K RAM. The *Music Machine* was written using a monochrome monitor. The colors were chosen to make the most pleasing monochrome display, but may need to be changed slightly for color monitors.

Program Operation

The *Music Machine* is controlled by a main menu that allows direct access to all of the program's functions (see Figure 2). The first menu selection allows the creation of a totally new song. When this option is chosen, the computer blanks the screen and locates a cursor at the upper left of the screen. This is the display for the creation of a song (referred to as a music table). In this mode the program works as a simple text editor. The cursor can be moved in any direction by the arrow keys, and the insert, delete and backspace keys all work as normal.

To create a music table is very simple.

Jerry Stajduhar is a field engineer and is currently working on his bachelor's degree in computer science.

The syntax for entering notes is exactly the same as that used for the BASIC PLAY statement. For example, to play the whole notes A, F and G you simply enter AFG on the music table. To indicate a half note, follow the note with a '2' (A2) — use 4, 8, 16, 32 or 64 for other note values. Sharps are indicated with a '#' and flats with a '-'. There are numerous different symbols the program recognizes; they are all listed under the PLAY command in the BASIC reference guide that comes with your computer. I will not list them here (the Tandy manual does a very good job of describing them), but I will mention a few warnings:

- The lowest note that can be played is 'A' in octave '0'.
- '>' can be used to raise the octave, but is not mentioned.
- '<' can be used to lower the octave.
- Each octave starts with 'C' and ends with 'B'.
- There are some combinations the computer can play but the program will not draw under menu option six (for example, 1/32 rest). This does not cause an error — it just won't be drawn.
- Use capital letters when creating the music table, otherwise the program will not draw the notes.
- The notes, sharps and flats, dots and values must be entered in a certain order if more than one is used on the same note. The order that works on my computer is: NOTE SHARP/FLAT VALUE DOT.
- This program recognizes a space as the end of the table. As soon as the program encounters a space, it ignores the rest of the table. Therefore, do not embed spaces in the music table.
- The music table is only one page long. There is no way to switch to another page to lengthen the song. The music table allows a maximum of 800 symbols.
- Typing '^' erases the entire table.
- If you have a very long song, the insert and delete functions may take a few seconds to work.
- The program saves all songs in the subdirectory \MUSIC\. This must be created on each disk the music program is on. To create the directory type MKDIR "\MUSIC" from the BASIC OK prompt. This must be done prior to using the program the first time.

Once you have created the table, press ENTER to exit to the main menu

(Figure 3 is a sample music table for the song "America the Beautiful").

The next menu option allows you to play the music table that is currently in memory. This selection has two options. If you enter a '2', the song is played immediately. If you enter a "-2" the music table is displayed while the song is being played. This is very useful if you are trying to debug a song since you can follow the notes that are being played.

If the program detects an invalid symbol in the music table, it will not play the song, instead it displays the message: ***ERROR DURING MUSIC PLAY***. If this occurs, you need to look through the music table to find what is wrong. As mentioned, the order of sharps and flats and other note modifiers is important. Periodically, BASIC appears to do nothing when you select option two. This is because the program requires a lot of string space to play the song, and BASIC must make the space available. The only solution is to wait the few seconds required until the program responds to the play option.

Menu options three, four and five are used to maintain disk files of the songs you have created. All three of these work the same way with simple prompts that are self-explanatory. If you accidentally select one of these options, you can exit by entering a capital X. They are all error trapped to protect against mistakes that could bomb the program and cause the music table to be lost.

Menu option six allows you to convert the music table to regular sheet music. When making this selection, you are asked a few questions that are used to set up the music sheet. The first query is ENTER FIRST POSITION. This is a number from zero to 800 that indicates the note the music sheet will start with. The program only prints one screen of music at a time. If your song requires multiple screens to display the entire song, then each screen has to be printed individually, starting at a different first position each time (when exiting this option, the computer displays the number of the last note it drew at the bottom of the screen).

The next prompt is ENTER NOTE SEPARATION. At this prompt, enter a number greater than or equal to zero. The larger this number, the more space there is between each note. This is useful to allow room to write words under the music if it is printed on a printer. This is also useful because the program does not draw the clef or separate the music into measures; this allows the space

needed to add these things.

The final question is DO YOU WANT PRINTOUT COLORS (Y/N). This question sets the background and foreground colors. If you have a CGP-220 printer, you need to use the printout colors, or you waste a lot of ink when the computer prints the screen. If you are planning to print the music, the screen dump routine that is included with the operating system has to be used. Using the screen dump requires that you load the graphics program before entering BASIC: From the DOS prompt, type GRAPHICS and press ENTER. Then, when you have the music sheet on the display (using the printout colors) press SHIFT PRINT. This copies the screen display to the printer. To return to the main menu press any key when the music sheet is finished being drawn on the screen.

The "Return to Table" selection (option seven) takes you back to the music table that is currently in memory so it can be edited. This selection works exactly like option one except it does not clear the table before returning to the text editing mode.

Option eight allows you to name the song you have created. This name is only used at the top of the screen when the music sheet is printed. The name can be up to 30 characters long. The default name is "not named." Option nine executes the complex BASIC statement — END.

Final Comments

The *Music Machine* is an enjoyable program to play with. It allows you to create music or listen to old favorites (though it is a little bit of work to convert the sheet music to music table symbology). The program should be able to accommodate any changes in BASIC that Tandy might come out with. The PLAY statement, which is the heart of this program, currently does not allow use of the multiple voices the computer can generate. But, if Tandy decides to include that in future versions of the language, the program should be able to use it (due to the free-format text editing mode).

Although I had a great time writing this program and I think it is fun to use, I have something to admit . . . I am not, and never have been, very good at music. I can barely read music and certainly cannot play an instrument. Therefore, there are probably things about this program that will irritate someone who is good at music. If I have made some glaring error, I hope one of the experts in the field will let me know! □

Figure 1: Sample Printout

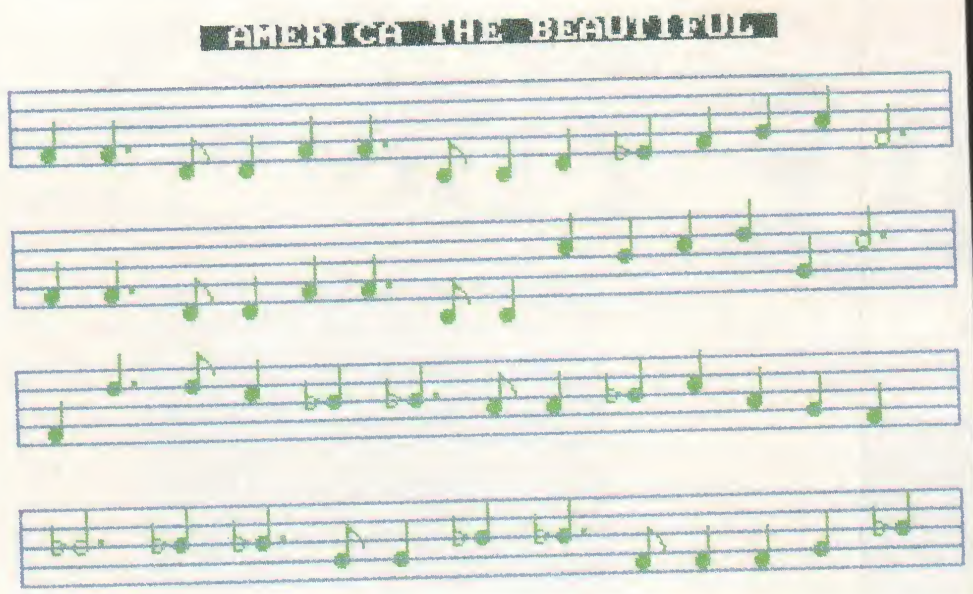


Figure 2: Main Menu

```
1 ) CREATE MUSIC TABLE
2 ) PLAY MUSIC TABLE
3 ) SAVE MUSIC TABLE
4 ) LOAD MUSIC TABLE
5 ) ERASE MUSIC FILE
6 ) DRAW MUSIC SHEET
7 ) RETURN TO TABLE
8 ) CHANGE NAME
9 ) END

?



NAME = AMERICA THE BEAUTIFUL
```

Figure 3: Music Table

```
F4F4.D8D4F4F4.C8C4D4E-4F4G4A4F2.F4F4.D8D
4F4F4.C8C4>C4<B4>C4D4<G4>C2.<F4>D4.D8C4<
B-4B-4.A8A4B-4>C4<A4G4F4B-2.B-4B-4.G8G4B
-4B-4.F8F4F4G4B-4F4>C4<B-2.
```

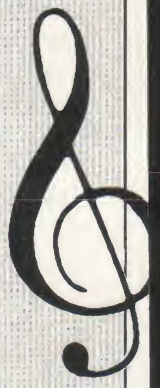
The listing:

```
10 REM ************************************************
20 REM **** THE MUSIC BOX    BY    JERRY STAJDUHAR  ****
30 REM **** FOR THE TANDY 1000 WITH 256K AND 1 DRIVE****
40 REM ************************************************
50 SCREEN 0
60 CLEAR ,,,32768!
70 SCREEN 5
80 CLS
90 REM **** MUSIC SYMBOL ARRAYS ****
100 DIM WHOLE(74),HALF(74),QUARTER(74),EIGHTH(74),SIXTEEN(74),SHARP(74),FLAT(74)
,REST(74),QREST(74),EREST(74),SREST(74),DOT(20)
110 KEY OFF
120 REM **** DEFAULT SONG NAME ****
130 MNAME$="NOT NAMED"
140 REM **** ASCII VALUES FOR ARROW KEYS ****
150 UP$=CHR$(0)+CHR$(&H48)
160 DOWN$=CHR$(0)+CHR$(&H50)
170 RIGH$=CHR$(0)+CHR$(&H4D)
180 LEF$=CHR$(0)+CHR$(&H4B)
190 REM **** ARRAY FOR NOTES ****
200 DIM MUSIC$(801)
210 REM **** MAIN MENU ****
220 LOCATE 1,1,0:PRINT"          THE MUSIC MACHINE"
230 FOR X=0 TO 800:MUSIC$(X)=" ":NEXT X:CLS
240 CLS:LOCATE 1,1,1:COLOR 15,0
250 LOCATE 20,1,1:PRINT "NAME = "MNAME$:LOCATE 1,1,1:COLOR 15,0
260 PRINT"1) CREATE MUSIC TABLE":PRINT"2) PLAY MUSIC TABLE "CHR$(241):PRINT"3) S
AVE MUSIC TABLE":PRINT "4) LOAD MUSIC TABLE":PRINT"5) ERASE MUSIC FILE":PRINT"6)
 DRAW MUSIC SHEET"
270 PRINT"7) RETURN TO TABLE":PRINT"8) CHANGE NAME":PRINT "9) END":PRINT
280 INPUT X:IF (X<=9 AND X>0) OR X=-2 THEN 290 ELSE CLS:GOTO 260
290 Z=0:Z1=0
300 ON ABS(X) GOTO 320,540,900,1020,1150,1440,1210,1910,1220
310 REM **** CREATE TABLE ****
320 CLS
330 X=0:WHILE MUSIC$(X)<>" ":MUSIC$(X)=" ":X=X+1:WEND
340 PRINT CHR$(177);:LOCATE 1,1,1
350 LOCATE 22,1:PRINT"ENTER = EXIT   ^ = ERASE":ROW=1:COL=1
360 LOCATE ROW,COL,1
370 REM **** SCAN KEYBOARD ****
380 KEYBD$=INKEY$
390 IF KEYBD$="" THEN 380 ELSE IF KEYBD$="^" THEN FOR X=0 TO 801:MUSIC$(X)=" ":N
EXT X:GOTO 320
400 REM **** ARROW KEYS ****
410 IF KEYBD$=LEF$ THEN MOVEMENT=1:IF COL>1 THEN COL=COL-1 ELSE IF ROW>1 THEN CO
L=40:ROW=ROW-1 ELSE COL=40:ROW=20
420 IF KEYBD$=RIGH$ THEN MOVEMENT=1:IF COL<40 THEN COL=COL+1 ELSE IF ROW<20 THEN
 COL=1:ROW=ROW+1 ELSE COL=1:ROW=1
430 IF KEYBD$=UP$ THEN MOVEMENT=1:IF ROW >1 THEN ROW=ROW-1 ELSE ROW=20
440 IF KEYBD$=DOWN$ THEN MOVEMENT=1:IF ROW<20 THEN ROW=ROW+1 ELSE ROW=1
450 REM **** DELETE/INSERT/ENTER/BACKSPACE ****
460 IF KEYBD$=CHR$(127) THEN X=((ROW-1)*40)+(COL-1):Z=ROW:Z1=COL:WHILE MUSIC$(X+
1)<>" ":MUSIC$(X)=MUSIC$(X+1):X=X+1:WEND:MUSIC$(X)=" ":CLS:LOCATE 1,1,1:GOTO 121
0
470 IF KEYBD$=CHR$(18) THEN X=((ROW-1)*40)+(COL-1):MUSIC$(800)=" ":Z=ROW:Z1=COL:
Y=X:WHILE MUSIC$(X)<>" ":X=X+1:WEND:WHILE X=>Y:MUSIC$(X+1)=MUSIC$(X):X=X-1:WEND:
```

```
MUSIC$(Y)="*":CLS:MUSIC$(8ØØ)=" ":LOCATE 1,1,1:GOTO 121Ø
48Ø IF KEYBD$=CHR$(13) THEN CLS:GOTO 25Ø
49Ø IF KEYBD$=CHR$(8) THEN LOCATE ROW,COL:PRINT " ";:MUSIC$(((ROW-1)*4Ø)+(COL-1)
)=" ":KEYBD$=LEF$:GOTO 41Ø
5ØØ LOCATE ROW,COL,1
51Ø IF MOVEMENT <>1 THEN MUSIC$(((ROW-1)*4Ø)+(COL-1))=KEYBD$:PRINT KEYBD$;:KEYBD
$=RIGH$:MOVEMENT=Ø:GOTO 41Ø ELSE MOVEMENT=Ø
52Ø GOTO 38Ø
53Ø REM **** PLAY MUSIC ****
54Ø A$="":B$="":C$="":D$="":FOUNDEND=Ø:Y=Ø:PB=Ø:PC=Ø:PD=Ø
55Ø IF X<Ø THEN X=Ø:CLS:LOCATE 1,1,Ø:WHILE MUSIC$(X)<>" ":PRINT MUSIC$(X);:X=X+1
:WEND ELSE LOCATE 22,1,Ø:PRINT"** PLAYING **"
56Ø REM **** BUILD STRINGS A,B,C,D OF 2ØØ CHARACTERS EACH ****
57Ø FOR X=Y TO 2ØØ
58Ø IF MUSIC$(X)=" " THEN FOUNDEND=1:X=2Ø1 ELSE A$=A$+MUSIC$(X)
59Ø NEXT X
6ØØ Y=2Ø1
61Ø IF FOUNDEND<>1 AND MUSIC$(Y)<>" " THEN IF MUSIC$(Y)<CHR$(65) OR MUSIC$(Y)>CH
R$(9Ø) THEN A$=A$+MUSIC$(Y):Y=Y+1:GOTO 61Ø
62Ø IF MUSIC$(Y)=" " THEN FOUNDEND=1
63Ø IF FOUNDEND=1 THEN 82Ø ELSE PB=1
64Ø FOR X=Y TO 4ØØ
65Ø IF MUSIC$(X)=" " THEN FOUNDEND=1:X=4Ø1 ELSE B$=B$+MUSIC$(X)
66Ø NEXT X
67Ø Y=4Ø1
68Ø IF FOUNDEND<>1 AND MUSIC$(Y)<>" " THEN IF MUSIC$(Y)<CHR$(65) OR MUSIC$(Y)>CH
R$(9Ø) THEN B$=B$+MUSIC$(Y):Y=Y+1:GOTO 68Ø
69Ø IF MUSIC$(Y)=" " THEN FOUNDEND=1
7ØØ IF FOUNDEND=1 THEN 82Ø ELSE PC=1
71Ø FOR X=Y TO 6ØØ
72Ø IF MUSIC$(X)=" " THEN FOUNDEND=1:X=6Ø1 ELSE C$=C$+MUSIC$(X)
73Ø NEXT X
74Ø Y=6Ø1
75Ø IF FOUNDEND<>1 AND MUSIC$(Y)<>" " THEN IF MUSIC$(Y)<CHR$(65) OR MUSIC$(Y)>CH
R$(9Ø) THEN C$=C$+MUSIC$(Y):Y=Y+1:GOTO 75Ø
76Ø IF MUSIC$(Y)=" " THEN FOUNDEND=1
77Ø IF FOUNDEND=1 THEN 82Ø ELSE PD=1
78Ø FOR X=Y TO 8ØØ
79Ø IF MUSIC$(X)=" " THEN FOUNDEND=1:X=8Ø1 ELSE D$=D$+MUSIC$(X)
8ØØ NEXT X
81Ø REM **** SET ERROR TRAP ****
82Ø ON ERROR GOTO 88Ø
83Ø REM **** PLAY MUSIC STRINGS ****
84Ø PLAY A$:IF PB=1 THEN PLAY B$:IF PC=1 THEN PLAY C$:IF PD=1 THEN PLAY D$
85Ø ON ERROR GOTO Ø
86Ø CLS:GOTO 25Ø
87Ø REM **** ERROR DURING PLAY ****
88Ø PRINT:PRINT"***ERROR DURING MUSIC PLAY***":PRINT"***INVALID CHARACTER***":IN
PUT "PRESS ENTER TO CONTINUE";A$:CLS:LOCATE 1,1,1:RESUME 121Ø
89Ø REM **** SAVE FILE UNDER SUB-DIRECTORY "MUSIC" ****
9ØØ CLS:PRINT"ENTER FILE NAME TO SAVE THIS UNDER      (X=EXIT)":INPUT A$:IF A$="
X" THEN CLS:GOTO 26Ø
91Ø ON ERROR GOTO 1ØØØ
92Ø A$="MUSIC\"+A$:PRINT A$:OPEN "O",1,A$
93Ø X=Ø
94Ø WHILE MUSIC$(X)<>" "
95Ø PRINT #1,CHR$(34);MUSIC$(X);CHR$(34);
96Ø X=X+1:WEND
97Ø CLOSE 1
```

```
98Ø ON ERROR GOTO Ø
99Ø CLS:GOTO 26Ø
1ØØØ PRINT"ERROR HAS OCCURED DURING WRITE":INPUT "PRESS ENTER TO CONTINUE";A$:CL
OSE:RESUME 9ØØ
1Ø1Ø REM **** LOAD FILE FROM DISK ****
1Ø2Ø CLS:FILES "\MUSIC\":PRINT:PRINT"ENTER FILE TO LOAD (X=EXIT)":INPUT A$:IF A$
="X" THEN CLS:GOTO 24Ø
1Ø3Ø FOR X=Ø TO 8ØØ:MUSIC$(X)=" ":NEXT X
1Ø4Ø ON ERROR GOTO 113Ø
1Ø5Ø A$="MUSIC\"+A$:X=Ø
1Ø6Ø OPEN "I",1,A$
1Ø7Ø WHILE NOT EOF(1)
1Ø8Ø INPUT #1,MUSIC$(X)
1Ø9Ø X=X+1
11ØØ WEND
111Ø CLOSE #1:MNAME$=RIGHT$(A$,(LEN(A$)-6))
112Ø ON ERROR GOTO Ø:GOTO 24Ø
113Ø CLS:PRINT"ERROR OCCURED DURING READ":INPUT"PRESS ENTER TO CONTINUE";A$:RESU
ME 1Ø2Ø
114Ø REM **** ERASE A FILE FROM DISK ****
115Ø CLS:FILES "\MUSIC\":PRINT:PRINT"ENTER FILE TO ERASE (X=EXIT)":INPUT A$
116Ø IF A$="X" THEN CLS:GOTO 25Ø
117Ø ON ERROR GOTO 12ØØ
118Ø A$="MUSIC\"+A$:KILL A$
119Ø ON ERROR GOTO Ø:CLS:GOTO 25Ø
12ØØ PRINT"***FILE NOT FOUND***":INPUT "PRESS ENTER TO CONTINUE";A$:RESUME 115Ø
121Ø ON ERROR GOTO Ø:CLS:X=Ø:LOCATE 1,1:WHILE MUSIC$(X)<>" ":PRINT MUSIC$(X);:X=
X+1:WEND:IF Z*Z1=Ø THEN GOTO 35Ø ELSE ROW=Z:COL=Z1:Z=Ø:Z1=Ø:GOTO 36Ø
122Ø END
123Ø REM **** SUBROUTINE TO CREATE THE NOTES AND STORE THEN IN ARRAYS ****
124Ø CIRCLE (1ØØ,1ØØ),2
125Ø GET (98,89)-(1Ø8,1Ø2),WHOLE
126Ø LINE (1Ø2,1ØØ)-(1Ø2,89)
127Ø GET (98,89)-(1Ø8,1Ø2),HALF
128Ø PAINT (1ØØ,1ØØ)
129Ø GET (98,89)-(1Ø8,1Ø2),QUARTER
13ØØ X=1Ø3:Y=91:GOSUB 14ØØ:GET (98,89)-(1Ø8,1Ø2),EIGHTH
131Ø X=1Ø3:Y=94:GOSUB 14ØØ:GET (98,89)-(1Ø8,1Ø2),SIXTEEN
132Ø CLS:LINE(1ØØ,95)-(1ØØ,1Ø5):LINE(1Ø4,95)-(1Ø4,1Ø5):LINE(98,99)-(1Ø6,97):LINE
(98,1Ø3)-(1Ø6,1Ø1):GET(98,95)-(1Ø8,1Ø5),SHARP
133Ø CLS:PSET(99,1Ø2):PSET(1ØØ,1Ø2):PSET(1Ø1,1Ø1):PSET(1Ø2,1ØØ):PSET(1Ø2,99):PSE
T(1Ø1,99):PSET(1ØØ,98):PSET(99,98):LINE(98,1Ø2)-(98,93):GET(98,89)-(1Ø8,1Ø2),FLA
T
134Ø CLS:LINE(98,99)-(1Ø2,1Ø1),,BF:GET (98,89)-(1Ø8,1Ø2),REST
135Ø CLS:PSET(98,9Ø):PSET(99,91):PSET(1ØØ,92):PSET(99,93):PSET(98,94):PSET(99,95
):PSET(1ØØ,96):PSET(1Ø1,97):PSET(1ØØ,97):PSET(99,97):PSET(98,98):PSET(99,99):GET
(98,89)-(1Ø8,1Ø2),QREST
136Ø CLS:LINE (1Ø5,93)-(1Ø5,1Ø1):X=1Ø1:Y=95:GOSUB 142Ø:GET(98,89)-(1Ø8,1Ø2),ERES
T
137Ø X=1Ø1:Y=99:GOSUB 142Ø:GET (98,89)-(1Ø8,1Ø2),SREST
138Ø CLS:PSET(1ØØ,1ØØ):PSET(1Ø1,1ØØ):PSET(1ØØ,1Ø1):PSET(1Ø1,1Ø1):GET(98,98)-(1Ø1
,1Ø1),DOT
139Ø RETURN
14ØØ PSET(X,Y):PSET(X,Y-1):PSET(X+1,Y):PSET(X+2,Y+1):PSET(X+3,Y+2):PSET(X+3,Y+3)
:PSET(X+4,Y+4):PSET(X+4,Y+5)
141Ø RETURN
142Ø CIRCLE (X,Y),1:PSET(X+1,Y):PSET(X+2,Y):PSET(X+3,Y-1):RETURN
143Ø REM **** DRAW THE SHEET ****
144Ø Z1=1:CLS:PRINT"ENTER FIRST POSITION":INPUT Z1:IF Z1>Ø THEN Z1=Z1-1
145Ø SEP=Ø:PRINT"ENTER NOTE SEPERATION":INPUT SEP
```

```
146Ø INPUT "DO YOU WANT PRINT-OUT COLORS (Y/N)";A$:IF A$<>"Y" THEN A$="N"
147Ø NLEN=2Ø-(LEN(MNAME$)/2):LOCATE 1Ø,1,Ø
148Ø PRINT"              WORKING":IF A$="N" THEN COLOR 15,Ø:GOSUB 124Ø:CLS:COLOR 9
,Ø ELSE COLOR 5,15:CLS:GOSUB 124Ø:CLS:COLOR 1,15:PAINT (1Ø,1Ø),15:LOCATE 1,NLEN,
Ø:COLOR 15,1:PRINT " "MNAME$" ";:COLOR 1,15
149Ø LENGTH=31Ø:MAXX=295
15ØØ IF A$="N" THEN GOSUB 188Ø
151Ø LOCATE 1,1,Ø::X=2Ø:Y=44:T=Z1:OFFSET=Ø:POSN=Ø:IF A$="N" THEN PRINT TAB(NLEN)
;MNAME$
152Ø REM **** LOOP TO POSITION THE NOTES ****
153Ø WHILE MUSIC$(T)<>" "
154Ø IF X>MAXX THEN X=2Ø:Y=Y+45:IF Y>2ØØ THEN TEMP$=MUSIC$(T):FLG=1:MUSIC$(T)="
":GOTO 16ØØ
155Ø IF MUSIC$(T)>="A" AND MUSIC$(T)<="G" THEN GOSUB 164Ø:GOTO 16ØØ
156Ø IF MUSIC$(T)="P" THEN GOSUB 181Ø:GOTO 16ØØ
157Ø IF MUSIC$(T)="<" THEN OFFSET=OFFSET+21:T=T+1:GOTO 16ØØ
158Ø IF MUSIC$(T)=">" THEN OFFSET=OFFSET-21:T=T+1:GOTO 16ØØ
159Ø T=T+1
16ØØ WEND:IF FLG=1 THEN FLG=Ø:MUSIC$(T)=TEMP$
161Ø IF A$="Y" THEN GOSUB 188Ø
162Ø WHILE INKEY$="":WEND:COLOR 15,Ø:CLS:LOCATE 25,1:PRINT STRING$(4Ø,32);:LOCAT
E 18,1,1:PRINT "END ="T:GOTO 25Ø
163Ø REM **** POSITION OF INDIVIDUAL NOTES ****
164Ø IF MUSIC$(T)="C" THEN POSN=5
165Ø IF MUSIC$(T)="D" THEN POSN=8
166Ø IF MUSIC$(T)="E" THEN POSN=11
167Ø IF MUSIC$(T)="F" THEN POSN=14
168Ø IF MUSIC$(T)="G" THEN POSN=17
169Ø IF MUSIC$(T)="A" THEN POSN=2Ø
17ØØ IF MUSIC$(T)="B" THEN POSN=23
171Ø REM **** SPECIAL SHAPES AND FRACTIONAL VALUES ****
172Ø IF MUSIC$(T+1)="#" THEN PUT(X,Y+OFFSET-POSN+6),SHARP,XOR:T=T+1:X=X+11
173Ø IF MUSIC$(T+1)="-" THEN PUT(X,Y+OFFSET-POSN),FLAT,XOR:T=T+1:X=X+7
174Ø IF MUSIC$(T+1)="2" THEN PUT(X,Y+OFFSET-POSN),HALF,XOR:X=X+12+SEP:T=T+2:DC=-
4:GOTO 179Ø
175Ø IF MUSIC$(T+1)="4" THEN PUT(X,Y+OFFSET-POSN),QUARTER,XOR:X=X+12+SEP:T=T+2:D
C=-4::GOTO 179Ø
176Ø IF MUSIC$(T+1)="8" THEN PUT(X,Y+OFFSET-POSN),EIGHTH,XOR:X=X+12+SEP:T=T+2:DC
=Ø:GOTO 179Ø
177Ø IF MUSIC$(T+1)="1" AND MUSIC$(T+2)="6" THEN PUT(X,Y+OFFSET-POSN),SIXTEEN,XO
R:X=X+12+SEP:T=T+3:DC=Ø:GOTO 179Ø
178Ø PUT(X,Y+OFFSET-POSN),WHOLE,XOR:T=T+1:DC=-2:X=X+9+SEP
179Ø IF MUSIC$(T)="." THEN PUT(X-2+DC-SEP,Y+OFFSET-POSN+7),DOT,XOR:X=X+6:T=T+1
18ØØ RETURN
181Ø IF MUSIC$(T+1)="2" THEN PUT (X+2,Y-25),REST,XOR:X=X+12+SEP:T=T+2:GOTO 186Ø
182Ø IF MUSIC$(T+1)="4" THEN PUT (X+2,Y-19),QREST,XOR:X=X+1Ø+SEP:T=T+2:GOTO 186Ø
183Ø IF MUSIC$(T+1)="8" THEN PUT (X,Y-2Ø),EREST,XOR:X=X+12+SEP:T=T+2:GOTO 186Ø
184Ø IF MUSIC$(T+1)="1" AND MUSIC$(T+2)="6" THEN PUT (X,Y-2Ø),SREST,XOR:X=X+12+S
EP:T=T+3:GOTO 186Ø
185Ø PUT(X+2,Y-27),REST,XOR:T=T+1:X=X+12+SEP
186Ø IF MUSIC$(T)="." THEN PUT(X-4-SEP,Y-16),DOT,XOR:X=X+3:T=T+1
187Ø RETURN
188Ø FOR Y=2Ø TO 17Ø STEP 45:FOR Z=Ø TO 24 STEP 6:LINE(1Ø,Y+Z)-(LENGTH,Y+Z):NEXT
 Z:LINE(1Ø,Y+Z-6)-(1Ø,Y+Z-29):LINE(LENGTH,Y+Z-6)-(LENGTH,Y+Z-29):NEXT Y
189Ø RETURN
19ØØ REM **** CHANGE SONG NAME ****
191Ø CLS:PRINT
192Ø INPUT"ENTER NEW NAME OR X TO EXIT (MAX 3Ø CHR)";A$:IF A$="X" THEN 24Ø ELSE
IF LEN(A$)>3Ø THEN PRINT "ERROR":BEEP:GOTO 192Ø ELSE MNAME$=A$:GOTO 24Ø
```



Dispelling some of the mysteries and misconceptions of IBM compatibility

What Makes a Compatible?

By John B. Harrell, III
PCM Contributing Editor

I had an interesting experience not long ago. I logged into my favorite Special Interest Group on CompuServe only to be greeted with disparaging remarks concerning my recent column on "Superfast Displays" (writing directly into video memory). It seems the subject of contention was my promoting direct hardware access methods while sacrificing machine compatibility.

In fact, I explicitly stated in the article that this method should not become widespread practice but should be limited to specific applications. I work every day with IBM PCs, Wangs, DEC Rainbows, Zeniths and some KayPros (with MS-DOS option).

These machines are just about as incompatible a group as could possibly be imagined. To get the most from your software investment under these circumstances, you must understand techniques such as those I presented and the differences between the various computers. This column is devoted to exploring this broad area of interest with examples and programming tips.

John B. Harrell, III has written for microcomputer magazines for three years. He holds a bachelor's degree in computer science and is a software technical expert for Navy electronic support measures systems.

The Definition of Compatible

There are many different meanings to the word "compatible" in reference to computers. The definition used by most people rates other brands of computers by the amount and diversity of software written specifically for the IBM PC that the other computer will run without modification. SubLogic's *Flight Simulator* and *Lotus 1-2-3* are key examples of compatibility tests in this area. If your computer can run these two programs, it will probably run most of the major software.

The other main area is generally called "data compatible," meaning the computer can use disks written in the standard IBM/MS-DOS format. For example, programs such as *MultiMate* for the Tandy 2000 can read data disks prepared with the IBM PC version of the program with no difficulty. Another example is *AutoCAD*, which relies heavily on the graphics capability of the target machine yet allows easy transfer of the data files to any other machine capable of running *AutoCAD*.

One machine, the DEC Rainbow, uses a non-standard disk format and effectively limits this data transportability. Even the Tandy 2000 suffers from this failing. While it fully supports the 40-track disk format, the 80-track Tandy 2000 disk drive, with its very thin heads, it might have difficulty creating a disk that can be used on an IBM PC. You must first bulk-erase the disk and use PCMAKER to format it, and both disk drives must be properly aligned or the disk will not be usable in a 40-track disk drive.

Differing degrees of software compatibility can be achieved by using varied programming techniques. The MS-DOS (PC-DOS) operating system provides a rich programming environment. Figure 1 summarizes the more than 70 functions provided in Version 2.xx for handling files, installing interrupt drivers, requesting system services and many more. The newer versions (MS-DOS 3.xx for the Tandy 3000) provide even more to support networking and shared resources.

The MS-DOS part of the system also provides functions for accessing the disk drives directly (absolute disk read/write), making resident software and error handling. These many features are well-documented and supported on all machines. Furthermore, Microsoft has maintained excellent upward mobility in recent releases of MS-DOS.

The most important concept to grasp here is this: If you use only "generic" MS-DOS operating system features, your software will run on all machines that use this operating system. This is true even for systems that are heavily tailored by the developer, such as the

Wang PC. It would seem Wang has gone out of its way to make their MS-DOS implementation hard to use, but the basic functions are all intact.

Using only the generic system calls costs plenty. The most obvious losses are controlling the video display and ability to perform sophisticated copy protection schemes by manipulating the disk controller. You can regain some of this loss by using the extended video and keyboard commands provided by ANSI.SYS. But, here again, there is a penalty — video display speed decreases significantly.

The program fragment in Figure 2 is a brief example of the complexity required for using ANSI.SYS function calls. The short PASCAL procedure mimics *Turbo* PASCAL's GotoXY procedure by positioning the cursor using ANSI.SYS control codes. To use these codes, the device driver must be installed by adding the line DEVICE=ANSI.SYS to your configuration file (CONFIG.SYS).

The Next Level

As I explained in my last column, there are three major divisions of the operating system. Your closest contact is with the DOS command processor contained in the file COMMAND.COM. This program provides the MS-DOS user interface, allowing communication with the operating system. It is also the source of the embedded commands such as COPY, DIR, etc.

Below the command processor is the MS-DOS operating system itself. This provides the many functions that are summarized in Figure 1 and is the interface to the Basic Input/Output System (BIOS) code routines which talk to the hardware directly.

In an IBM compatible computer, the BIOS code is partly contained in a disk file on each of your disks and partly in the computer's Read-Only Memory (ROM). The code in the disk file extends the ROM code routines. The BIOS code design allows flexible extensions through interrupt service modifications, installable device drivers to correct problems and oversights and allows for easy additions of new technology peripheral equipment.

The next level of software programmer support resides in the Interrupt Service Routines contained in the BIOS code. Services in the 8088/80186 computers are requested by interrupts to the central processor. These interrupts can originate with the hardware or

can be programmed under software control.

Many of these software interrupt functions are duplicated by MS-DOS function calls and I encourage you to use those functions for long-range compatibility. Figure 3 provides a brief summary of the BIOS service routines provided in an IBM PC compatible BIOS and the code segment in Figure 4 is equivalent to the one in Figure 2.

Of the software functions listed in Figure 3, two are vital to programmers. The most used function is the video display interrupt followed closely by keyboard interrupt. Inadequate BIOS services are where many of the MS-DOS machines lose compatibility with the IBM PC "standard." For example, Wang does not document how they implemented these two vital services, and software expecting to use these routine services *will not run* and probably will force you to reset the computer. A similar situation exists on the Zenith computer systems (Z-100/120).

Of the incompatible machines, the Tandy 2000 BIOS code is remarkably complete. Most of the BIOS services are functionally identical to the IBM PC and I use my MS/PC-DOS programmer's reference manuals interchangeably. I even use the documented IBM BIOS source code listing to trace through the Tandy 2000 BIOS code.

Where services are not available, it is usually because the functions are generally hardware limited, but I'll delve more into this later. By the way, the IBM programmer's and hardware reference manuals make excellent references for your Tandy equipment.

The ultimate speed advantage is achieved when the hardware is accessed directly. Why do you have to pay the penalty for the disk operating system's overhead and required maintenance? The answer is simple — the operating system is the "coordinator" for every application that your computer runs and must perform many "housekeeping" chores to maintain control.

By achieving this great speed advantage, you have needlessly limited yourself to a very narrow range of installations and, don't forget, *the manufacturer is free to change the hardware configuration at any time*! For example, IBM does not guarantee they will maintain video display RAM in the same location — they have done so to date but might change in the future.

This latter method of programming has another serious side effect. With the

advent of *Windows, Top View* and other related software, "well-behaved" programs, i.e., programs that do not take advantage of the computer architecture, are a must. These systems cannot work as they have been designed if they cannot predict the behavior of the "child" processes they execute. In this case, the multitasking systems stop all other processes and give full control to the offending program.

Comparing Computers

Let's take a look at some of the differences between the IBM compatibles and the Tandy 2000 as an example of different hardware architectures. There are some differences I will not mention — I will be examining the differences from a programmer's point of view and some of the differences are simply cosmetic.

Just looking at them you immediately notice the keyboard — 83 keys on the IBM compatibles (except for the Tandy 1000) and 90 on the Tandy 2000. A closer inspection reveals that they have some *different* keys. Every key generates a "scan code" when it is pressed. Because of the obvious differences, the two computers have variations in the generated scan codes.

Fortunately, similar keys generate similar codes, but there are some limitations. For example, the Tandy 2000 does not have a Scroll Lock key, the plus (+) and minus (-) keys on the numeric keyboard, and you must contend with the additional codes for the F11 and F12 keys. Also, because the arrow keys are separate from the numeric keypad, using these keys in conjunction with the SHIFT and ALT keys now has significance.

Probably the most significant "improvement" in the Tandy 2000 is the video hardware. The controller is totally different from the IBM compatible computers and, while some operating modes are similar, the actual implementation is *very* different. The video controller's registers are mapped into I/O ports in the computer's address space — these ports are totally different on the two machines.

In the IBM compatibles, the video RAM is located a fixed segment address for both the monochrome (B000H) and color adapters (B800H). This video RAM is organized as a byte representing the character value to be displayed followed by a byte telling the video controller how to display the character (color underlining, blinking, etc.).

In the color graphics system, 16K of memory is available in the graphics mode and may be organized into four separate video pages of 80-character-wide lines. Using appropriate BIOS functions, one of these video pages can be displayed while you are writing to any of the other three pages. A simple instruction can then switch the active pages for an "instant" screen display.

When the PC compatible displays graphics, the video RAM is used to hold the graphics pixel data. In the 320 by 200 mode, two bits are used to represent a pixel on the screen yielding four colors: background or any of the other three palette selections. In the 640 by 200 mode, each bit in RAM represents a pixel location on the screen so pixels may be either off (background) or on (current foreground color).

Data is taken successively from the video RAM using 80 bytes as described above to represent one scan line on the video screen. Odd scan lines are taken from the first half of video memory and even scan lines are taken from the second half. Text displayed on the video screen in the graphics mode is written pixel by pixel from a table of characters stored in memory.

In the Tandy 2000, the monochrome RAM is managed similarly to the IBM-PC *with two exceptions* — both are very important. First of all, the attribute bytes are totally different from those described in my previous column. Second, the video RAM segment address "floats" at the upper end of the memory you have installed. You must first find out the size of memory installed and then calculate the video segment address.

The PASCAL program presented in "Superfast Displays" (January 1986) gives a good example of this technique. In fact, the only changes you must make in the program to make it run on the Tandy 2000 affect the video segment addresses and the attributes. Once these changes are made, the code in the other routines is capable of displaying data *with no other changes.*

Handling the color and graphics displays on the Tandy 2000 is another story. The 2000's graphics RAM begins at segment address E000H and extends for 32K. For 640 by 400 resolution, there are 256,000 distinct pixels to be

displayed. Each byte of the RAM in this area represents eight pixels across the line and, unlike the compatibles, bytes are taken from contiguous locations for even and odd scan lines on the video display.

To represent the full eight colors, the Tandy 2000 uses three bits of data. There are *three* distinct planes of memory all mapped into the segment space beginning at E000H and you control which one is active by using one of the ports. When you want to change one pixel location, you have to set or reset the corresponding bit in each of the three planes of memory to make the colors come out right.

The Tandy 2000 is a true bit-mapped graphics machine. Unlike the compatibles, all text and graphics are written to the display RAM in the full graphics mode. The 2000's BIOS code is remarkably compatible; when in the color text mode, writing to the screen using the normal IBM attributes and BIOS calls produces the correct display.

The single exception is that you cannot make characters blink by setting the appropriate attribute bit. The Tandy 2000 uses the values written in the three graphics planes to address palette registers on the video controller. These palette values are set by writing colors to different ports in the computer's address space. Just think of the effort required to draw the letter 'A' in white on a blue background — each pixel of the eight by eight character array must be set in each of the planes to achieve the proper colors.

The organization of the video RAM on the 2000 has several side effects. First of all, it is the reason the screen scrolls so slowly in the color text mode. Unlike the IBM PC, which has to move only 4K of RAM to scroll the screen, the Tandy 2000 has to shift 96K of RAM to produce the same scrolling effect.

Another side effect is the lack of separate video pages similar to the compatibles. In the true bit-mapped configuration, the Tandy 2000 uses all of the video RAM all of the time in the color mode. This is similar to the IBM PC's operation in the graphics mode where all of the 16K of video RAM is used to display the graphics picture.

Another area of incompatibility is generating sounds from the speaker. Both the IBM compatibles and the Tandy 2000 gate the output from an 8253-5 programmable timer chip to the speaker to make sound. The port address on the IBM compatibles for the chip control word is 43H and for the counter register data, 42H. The output is gated to the speaker by using port 61H.

On the Tandy 2000, the corresponding ports are 46H and 40H for the programmable timer and 00H for the speaker gate. The IBM timer chip ports are not too damaging as they correspond to values used by the Tandy 2000's timer. The speaker gate value is more of a problem as this is in the area of the programmable interrupt controller, and it can cause problems.

Furthermore, any IBM program that accesses port zero can drastically affect the behavior of the Tandy 2000. This port controls the speaker gate on the 2000. More importantly, it also controls the dynamic memory refresh updating and power to the keyboard. And to top it off, even if you overcome all of this, the Tandy 2000 operates the timer chip at a different frequency than the IBM PC, so the sounds produced have a higher pitch.

Two other areas of hardware incompatibility that might annoy programmers are the keyboard port accesses and the asynchronous communications ports. While their behavior is almost identical to the compatibles, the port addresses are totally different on the Tandy 2000.

What's more, Tandy changed the BIOS code structure to replace the more familiar IBM interrupt for the keyboard with one in a totally unexpected location (INT 9 is replaced by INT 78H on the 2000). The keyboard buffer (and the whole BIOS communication "page" in low address memory) is completely reorganized into different addresses. This does have some significant advantages — the Tandy 2000 has a 512-keystroke type-ahead buffer.

Conclusion

In the short space of this column, I have attempted to examine some of the things that make computers incompatible with another. A detailed discussion of the differences on, for example, the Tandy 2000 would be enough to fill a good-sized book. That should not deter you from being interested.

When I bought the *Turbo Editor Toolbox* from Borland International, I had to argue with them about getting the IBM PC version instead of a generic MS-DOS version. I was certain I could make the text editors run on the Tandy 2000 with a minimum of effort.

The changes I made to *MicroStar* are almost identical to the code I previously published in "Superfast Displays," and it is a super text processor for the Tandy 2000. In fact, I even added another couple of lines to look at the Boot ROM chips and determine what type of computer I am on. Now, the same program runs on any of the IBM compatibles *and* the Tandy 2000 with no changes. It's too bad Borland has not followed this line of programming — it sure would be nice to have *SideKick* and *SuperKey* for the Tandy 2000.

The moral of this story is this: You can convert software to run on almost any other machine with the proper understanding of the machines. You also need lots of patience. The source code is preferable, but not totally necessary. Several of us have patched IBM PC *Turbo* PASCAL to run on the Tandy 2000, including using the high resolution graphics modes.

Most commercial software is so shrouded in secrecy that I find the task unworthy. If I really want *Lotus 1-2-3*, I'll buy the Tandy 2000 version of it. You might also break some of the licensing agreements by disassembling the code to make your changes. Most of the popular software packages are available for the Tandy 2000.

If you have more than a passing interest in your computer's architecture and how to program for special features, I highly recommend *The Peter Norton Programmer's Guide to the IBM PC*. Other references that may help are the DOS and hardware reference manuals for the IBM PC and the technical manuals available from Tandy for your computer.

I hope this has provided some insight into the myriad differences in computer hardware that can affect compatibility. Several of you have asked me to write about how to make "Joe's Special Framis" software run on the Tandy 2000. The answer is simple — there really is no way other than to have an accomplished programmer *familiar with the hardware* make the changes.

My experimentation and nosiness has allowed me the opportunity to accomplish some of these changes. But the effort has often been far from rewarding and, often, very frustrating. Earlier, I mentioned patches for *Turbo* PASCAL that allow the IBM PC compiler to operate on the Tandy 2000. They are public domain patches and, if you have not received them yet, send me an SASE and I'll send you a copy of them. My address is 1519 A Carswell Circle, Bolling AFB, Washington, DC 20336.

□

Figure 1
MS-DOS Versions 2.xx and 3.xx Function Calls

Fcn #	Description
00H	Program Terminate
01H	Keyboard Input
02H	Display Output
03H	Auxiliary Input
04H	Auxiliary Output
05H	Printer Output
06H	Direct Console I/O
07H	Direct Console Input Without Echo
08H	Console Input Without Echo
09H	Print String
0AH	Buffered Keyboard Input
0BH	Check Standard Input Buffer (keyboard status)
0CH	Clear Keyboard Buffer, Invoke a Keyboard Function
0DH	Disk Reset
0EH	Select Disk
0FH	Open File
10H	Close File
11H	Search for First Entry
12H	Search for Next Entry
13H	Delete File
14H	Sequential Read
15H	Sequential Write
16H	Create File
17H	Rename File
18H	(internal DOS use)
19H	Return Current Disk
1AH	Set Disk Transfer Address
1BH	Allocation Table Information (IBM only)
1CH	Allocation Table Information for a Specific Device (IBM only)
1DH	(internal DOS use)
1EH	(internal DOS use)
1FH	(internal DOS use)
20H	(internal DOS use)
21H	Random Read
22H	Random Write
23H	File Size
24H	Set Relative Record Field
25H	Set Interrupt Vector
26H *	Create New Program Segment
27H	Random Block Read
28H	Random Block Write
29H	Parse Filename
2AH	Get Date
2BH	Set Date
2CH	Get Time
2DH	Set Time
2EH	Set/Reset Verify Switch
2FH	Get Disk Transfer Address

Fcn #	Description
30H	Get DOS Version Number
31H	Terminate Process and Remain Resident
32H	(internal DOS use)
33H	CTRL-BREAK Check
34H	(internal DOS use)
35H	Get Interrupt Vector
36H	Get Free Disk Space
37H	(internal DOS use)
38H	Set or Get Country Dependent Information
39H	Create Subdirectory
3AH	Remove Subdirectory
3BH	Change Current Subdirectory
3CH	Create a File Handle
3DH	Open a File Handle
3EH	Close a File Handle
3FH	Read From a File or Device
40H	Write to a File or Device
41H	Delete a File From a Specified Directory
42H	Move File Read/Write Pointer
43H	Change File Mode
44H	I/O Control for Devices
45H	Duplicate a File Handle
46H	Force a Duplicate of a File Handle
47H	Get Current Directory
48H	Allocate Memory
49H	Free Allocated Memory
4AH	Modify Allocated Memory Blocks
4BH	Load or Execute a Program
4CH	Terminate a Process
4DH	Get Return Code of a Subprocess
4EH	Find First Matching File
4FH	Find Next Matching File
50H	(internal DOS use)
51H	(internal DOS use)
52H	(internal DOS use)
53H	(internal DOS use)
54H	Get Verify Setting
55H	(internal DOS use)
56H	Rename a File
57H	Get/Set a File's Date and Time
58H	(internal DOS use)
59H *	Get Extended Error
5AH *	Create Temporary File
5BH *	Create New File
5CH *	Lock/Unlock File Access
5DH	(internal DOS use)
5EH	(internal DOS use)
5FH	(internal DOS use)
60H	(internal DOS use)
61H	(internal DOS use)
62H *	Get PSP Address

Note: All interrupt function calls marked with an asterisk (*) are functions added when Version 3.xx and higher were released.

Figure 2
Sample PASCAL Program Using Extended Video Control Codes

```
                                                    {$P256}
program test;
const
  teststr = 'XXX';
var
  i  : integer;
  ch : char;
{
  Locate the system cursor at column "x" in row "y" using
  an ANSI.SYS character string for extended video control
}
procedure gotoxy( x, y : integer );
var
  rowstr : string[2];
  colstr : string[2];
begin
  str(y+1,rowstr);      { convert row number to a string format   }
  str(x+1,colstr);      { convert column number to a string also  }
  write(#27'[',rowstr,';',colstr,'H'); { write ANSI.SYS sequence }
end; { gotoxy }

begin
  clrscr;
  for i := 0 to 24 do
    begin
      gotoxy(i*3,i);
      write(teststr);
    end;
  read(kbd,ch);
end.
```

Figure 3
BIOS Device I/O Services for the Tandy 2000

1) Keyboard Interrupt Functions
a. 0: Read Keyboard (destructive with wait)
b. 1: Scan Keyboard (non-destructive, no wait)
c. 2: Get current shift status
d. 3: Flush keyboard buffer
e. 4: Reset keyboard

2) Video Display Interrupt Functions
a. 0: Set CRT mode
b. 1: Set cursor type
c. 2: Set cursor position
d. 3: Get cursor position
e. 5: Select active page
f. 6: Scroll up
g. 7: Scroll down
h. 8: Read attribute/character
i. 9: Write attribute/character
j. 10: Write character only
k. 11: Set color palette
l. 12: Write pixel dot
m. 13: Read pixel dot
n. 14: Write in TTY mode
o. 15: Get CRT mode
p. 16: Get/Set character fonts
q. 17: Write attribute or color only
r. 18: Additional scroll functions

3) Serial Communications Interrupt
a. 0: Reset COM port
b. 1: Transmit character
c. 2: Receive character
d. 3: Get current COM status
e. 4: Flush COM buffer

4) Line Printer Interrupt
a. 0: Print character
b. 1: Reset printer port
c. 2: Get current printer status

5) System Clock Interrupt

a. 0: Get time of day
b. 1: Set time of day
c. 2: get date and time
d. 3: set date and time

6) Disk I/O Support Interrupt

a. 0: Reset specified disk
b. 1: Return status of last disk operation
c. 2: Read sector(s) from specified disk
d. 3: Write sector(s) to specified disk
e. 4: Verify sector(s) on specified disk
f. 5: Format track on the specified disk
g. 8: Return hard disk drive parameters
h. 9: Initialize hard disk drive parameters
i. 12: Perform seek on hard disk
j. 13: Reset hard disk system
k. 14: Read hard disk sector buffer
l. 15: Write hard disk sector buffer
m. 16: hard disk drive ready test

7) Boot Strap Interrupt

8) Print Screen Interrupt

9) System Equipment Status Interrupt

10) Memory Size Interrupt

Figure 4
Sample PASCAL Program Using BIOS Interrupt Function Calls

```
program test;
const
  teststr = 'XXX';
type
  regpack =
    record
      AX, BX, CX, DX, BP, SI, DI, DS, ES, Flags : integer;
    end;
var
  i  : integer;
  ch : char;
{
  Locate the system cursor at column "x" in row "y" using
  a BIOS interrupt 16 video call
}
procedure gotoxy( x, y : integer );
var
  regs : regpack;
begin
  with regs do
    begin
      AX := $Ø2ØØ;  { Set AH = 2 for "Set Cursor Position"      }
      BX := Ø;      { Set BH = current video page, assumed Ø    }
      DX := (y shl 8) + x;              { Set DH = row; DL = col }
      Intr(16,regs);              { Call BIOS code to set cursor }
    end
  end; { gotoxy }

begin
  clrscr;
  for i := Ø to 24 do
    begin
      gotoxy(i*3,i);
      write(teststr);
    end;
  read(kbd,ch);
end.
```

PCM

The advantages and how-to's of using tapes . . .

Coping with Cassettes

By F.H. Ingle

How is your Model 100 cassette interface working? For many, I suspect the answer is "seldom." Some of us have never tried out the cassette interface; others have tried it, but for a number of reasons found it unreliable. Still others have had success with using cassettes, but eased out of using them because they just seem to be more trouble than they're worth.

The purpose of this article is to show you how to use the cassette interface in such a way that it works first time — every time, and with a minimum of effort. We'll also look at how to set up a cassette library system, one that fits neatly into your briefcase, which will enable you to find what you are looking for in a matter of seconds.

Advantages of Cassettes

Before we get into the finer details of your Model 100 cassette interface, let's consider some reasons why using cassettes is advantageous. First, cassettes allow you to back up files on your Model 100 if they are inadvertently damaged. This could be caused by software that goes haywire, letting your batteries run down or your own nimble-fingered mishaps on the keyboard. If you ever install more memory in your system, or any of a number of other accessories that require a cold start, a backup of some kind is essential. If you are in the habit of backing up your files on cassette on a regular basis, you can take all of these problems in stride.

A second advantage is to enable you to stretch the memory capacity of the computer. By using cassettes, you can edit and print manuscripts and programs that are much longer than your memory can hold.

You can also maintain a library of programs ready to use on short notice. With so many nifty programs available, it just doesn't make sense to limit yourself to only the programs you can keep online in the 100.

With cassettes, you have the ability to exchange programs and text with other Model 100 users. One 90-minute cassette can hold 234K of text (that's around 100 typewritten pages), but can be mailed anywhere in the country for 56 cents. You'll have to admit that's quite a bargain!

Another advantage is that the cassette equipment is portable, just like your computer. The Model 100, a program library, a cassette recorder, spare batteries and miscellaneous cables all neatly fit into one briefcase (see Figure 1).

All of these advantages may be used as arguments for using a portable

Frank Ingle holds a master's degree in electrical engineering and is owner of Command Communications, a communications consulting firm. During 20 years' involvement with computers, he has invented two new computer systems and has done research for IBM, Honeywell and the Navy. He is also a member of the IEEE. Frank can be contacted at 2580 Park Street, Jacksonville, FL 32204.

GOING PORTABLE?

WHAT ARE THE TWO MOST IMPORTANT CONSIDERATIONS?

SIZE AND WEIGHT!

This printer is half the size and half the weight of your Model 100. Anywhere you happen to be, cruising at 30,000 feet or in a cab to an important business meeting, you can get a quick and quiet printout. A perfect companion for your Model 100/200/600.

FEATURES?

This printer has them all:

- 8½" wide thermal paper
- 512 byte buffer
- Epson MX-80 bit-mapped graphics
- Built-in automatic self test function
- Excellent User's Manual

(ACTUAL print sample)
80 Chars/Line, 40 CPS!
Enlarged!
underlined Emphasized!
CONDENSED gives you 160 chars. per line!

$99 GRAND TOTAL

You get the printer and a specially made 18" cable (connects printer to your Model 100/200/600 or NEC 8201/8401), 20' roll of paper, a paper holder, and 4 "C" batteries — everything you need to start printing immediately with the push of a button. This price also includes free UPS surface shipping in the cont. USA. There is no surcharge for VISA/MC/Amex — even the phone call is free.

RUGGED

This printer was designed to print over 300,000 lines without maintenance — that's over 104 pages every week for a year.

DIRECT TO YOU

We've been selling this printer to our portable customers and corporate accounts for over a year. Two of our largest accounts, NBC and National Geographic, use them in the field all over the world. To meet the demand we have gone direct to the Japanese manufacturer. Our large volume cost is unbelievable — that's why we can offer it direct to you at this fantastic price, yet still give you great service and support (6 month parts/labor warranty, 30 day money back satisfaction guarantee).

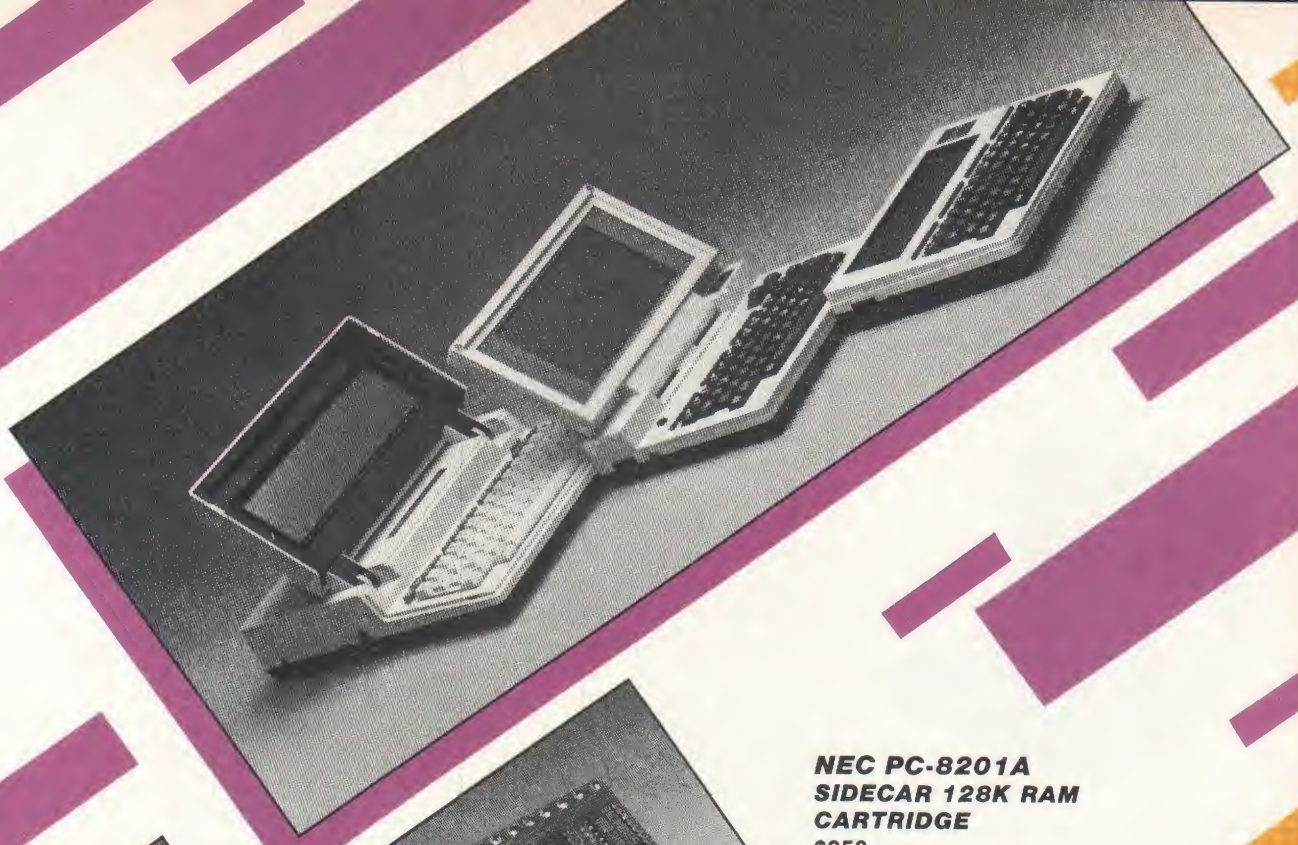
SUPPORT

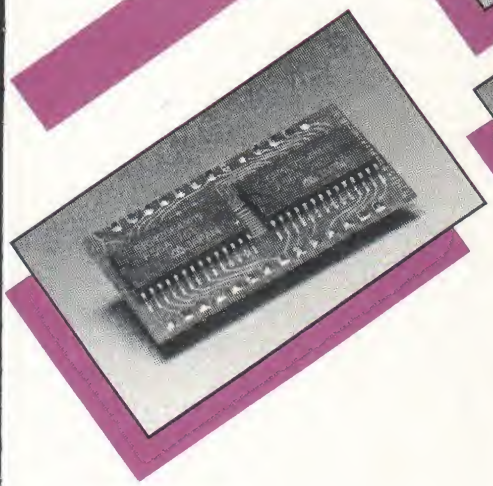
We stock paper (rolls and sheets), cables, and accessories for the printer. The full accessory kit costs only $24 and includes a carrying case, 100' roll of paper, and an A/C adapter (saves batteries when A/C power is convenient).

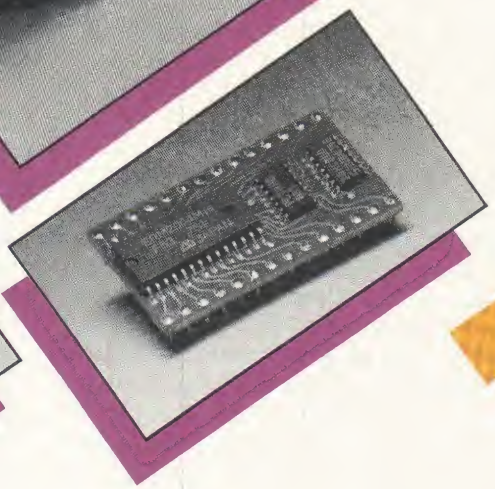
NO RISK

Try it for 30 days, ask your friends and business associates if they've ever seen anything like it. You can return it in 30 days for any reason and get a full refund. Don't let yourself miss this portable printer bargain of a lifetime.

$99







NEC PC-8201A SIDECAR 128K RAM CARTRIDGE

$259

Plugs into side expansion slot providing four additional banks of 32K each.

IT'S EASY TO ORDER

Send your order with payment to the address below. Or, if you prefer, credit card orders can be handled by phone—VISA, MasterCard, and American Express are welcome. California residents add 6% sales tax. Checks allow 3 weeks to clear. Institutions and Corporations call for purchase order approval.

1-800-732-5012 TOLL FREE

Orders only (8am. - 5pm. PST)

In California ***(805) 987-4788***

For orders or customer service

If you would like our latest product catalog or to find a dealer near you, please write or call (805) 987-4788

TANDY 200

24K Memory Module

$89 — two for $170

Can be installed in less than 2 minutes, the only tool required is a coin to pop open the hatch. The TANDY 200 is expandable to 72K by adding two of our 24K modules. Built with the lowest power and most reliable memory chips available. The pins are made specifically for the TANDY 200 chip socket. 2-Year Warranty.

MODEL 100

NEC PC-8201A, Olivetti M10

8K Memory Module

$29 — three for $79

Plugs into the socket provided—expanding your memory by 8,192 bytes. Super low power components for long battery life. Low profile which is very important to properly fit the NEC PC-8201A. Tens of thousands sold since 1983. 2-Year Warranty.

TANDY 600

96K Memory Unit

$249 — two for $469

NEW!

You can install one or two of these units inside your TANDY 600 — giving you a maximum of 224K.

PURPLE COMPUTING

420 Constitution Ave.
Camarillo, CA 93010
Telex: 888661 (PURPLE)

Canada: Canada Portable Computer, Langley, B.C. (604) 856-8858

RAM EXPANSIONS!

FOOLISH TO PAY MORE — DANGEROUS TO PAY LESS

floppy disk system as well. However, the most outstanding advantage of using cassettes can't be touched by any other system: economy. You can do all of these things for under $50. For those users who were attracted to the Model 100 because of its inexpensive price tag in the first place, this will probably be a deciding factor.

Figure 1

How Data Cassettes Work

Now with some convincing reasons that the cassette interface can be a useful tool, let's see how it works. How do those mystical beeps and burps that the cassette records get turned into meaningful data? You are, no doubt, already aware that all of the characters you see on the display of the Model 100 are stored in memory as binary numbers. Each character is assigned to a different number, which may be represented by a combination of eight binary digits (bits). The trick is, how can we record these "ones" and "zeros" on magnetic tape? A technique called Frequency Shift Keying (FSK) is used. Don't let the name scare you, though. The concept is really pretty simple. The Model 100 converts every zero to a sound whose pitch is 1200 Hz. That would sound something like playing the 'E' in the third octave above Middle 'C' on the piano. To represent a one, the Model 100 produces a sound whose pitch is 2400 Hz. That would be a note one octave higher on the piano. Those familiar with the inner workings of modems will recognize this as the same technique used to send data using the Bell-103 compatible modem, although the frequencies are different.

So that's it in a nutshell: The Model 100 turns ones and zeros into "music" so they can be recorded on cassettes. If you listen to a cassette recorded by the Model 100, you'll hear the 1200 Hz tone right after every pause. That's a string of zeros being written by the Model 100 to introduce the next section of text or program. Then you'll hear something like a buzz saw when actual data is being recorded. The "buzz saw" sound is caused by those two tones being turned on and off very quickly. If you listen to one second of tape, you will have heard 1,500 ones or zeros, or roughly 150 bytes of data (ASCII characters).

A little quick arithmetic would lead you to think 45K of data could be stored on five minutes worth of tape, however, this is not so. In addition to the data in the file, the Model 100 is also writing header records, synchronizing pulses, interblock gaps and checksums. All of these ensure the cassette system works smoothly and dependably, but they also take up additional space on the tape. It turns out that 13K of text can be recorded on one side of a 10-minute tape (i.e., in five minutes). If you have several files to record instead of one, plan on getting about 12K per side.

Why Won't All Recorders Work?

Now a very good question comes up. Why won't the Model 100 work with all cassette recorders? Why will it work with a $29.95 recorder and not my $1,000 stereo deck? One problem is the tone that represents a one, 2400 Hz, is towards the upper end of the capability of some recorders. Accordingly, the ones are not played back quite as loud as the zeros, and this confuses the Model 100.

If the cassette recorder has a tone control this may be easily corrected by adjusting it to the high (treble) side until the interface works dependably, and then always using that tone setting when using the recorder in conjunction with the Model 100. If a cassette recorder lacks a tone control, and doesn't seem to work with a Model 100, there is no simple way to make it work.

Another common problem that keeps cassette recorders from working with the Model 100 is signal level. The recorder must accept the signal exactly as it comes from the 100 because very few inexpensive recorders have a manual volume control, which affects the record level. If the recorder wants a louder or softer signal than the 100 furnishes, you're out of luck. On playback, though the level must also be just right, this level may be adjusted by turning the volume control to a setting that works dependably. Some recorders (including some very expensive ones) are not capable of handling or producing the required signal levels no matter what setting is used, and these are the ones that are of no use to Model 100 owners.

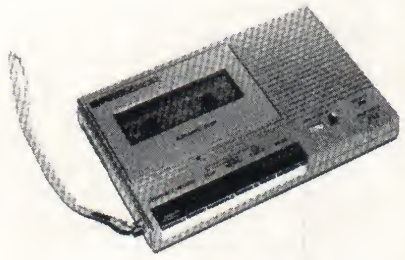
The best way to know if a particular model of cassette recorder will work with your Model 100 is to ask someone who has tried it. Be sure to ask for the best volume and tone settings as well. The Model 100 SIG on CompuServe is a good place to find out.

One recorder that works particularly well with the Model 100 is the Sears SR Series Model 560.26170250 (see Figure 2). You can find it in the 85/86 Office Equipment and Supplies Catalog on Page 49, catalog number 57 KX 21674. The current price is $39.99. (Note: The picture in the catalog is mislabeled — the correct picture is letter 'D'.) An identical recorder is also available from Radio Shack for $59.95. Ask for catalog number 14-812. In addition to the tone control, this model is smaller than the one normally offered by Radio Shack for use with the Model 100, and uses the same batteries as the 100. A tone control setting of '4' and a volume control setting of '7' on this unit gives dependable recordings every time.

Choosing the Right Tape

Once you have the right recorder, you'll need some cassette tapes. A tape salesperson will probably say you need the best tape for data recording. Though this may be good advice for other computers, it's not true for the Model 100's sophisticated cassette interface — it's not fussy about tape at all.

Figure 2



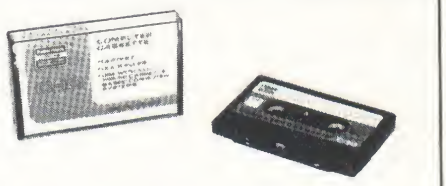
You can use just about any cassette tape with the Model 100. I suggest, however, that you use very short tapes to make finding files on the tapes easier. I recommend a 10-minute tape, which allows five minutes of data (about 12K) to be recorded on each side. This type of cassette is normally referred to as a "C-10."

If you can find these short tapes with

no leader, the work is even easier. The leader is the odd-colored tape spliced to both ends of most cassette tapes, which is stronger than regular tape, and is designed to handle the extra stress that is applied to the tape when it runs out in fast forward or rewind. The problem with most leaders is that you can't record on them, so you have to manually advance the tape past the leader before recording.

Radio Shack offers a certified data C-10 tape, catalog number 26-302, for $1.89 (see Figure 3). Both leaderless and leadered tapes are under the same catalog number. This is a bit of an overkill for our needs, but it's difficult to find less expensive C-10 tapes. If you prefer, look in the yellow pages under "tapes-sound-dealers" to find someone who sells tapes made for duplicator use. If they have their own cassette winding machine, they may be able to make tapes to order in any length and any quantity, and still beat Radio Shack's prices.

Figure 3



I suggest starting your library with about five tapes. If more are needed later you can add to it. While shopping, you may also want to pick up a head and capstan cleaning cassette (see Figure 4). Radio Shack offers one under catalog number 44-1163 for $4.99, or you could get a bottle of 90 percent isopropyl alcohol and a box of cotton swabs. Either of these can be used to keep the cassette recorder clean.

Paint Your Plug

Naturally, you'll need a cassette interface cable, which comes with your Model 100. Here's a hint that makes life simpler when using this cable: get some red fingernail polish and paint the larger gray plug. Then, put a drop of red nail polish on the recorder just above the MIC jack. This color coding scheme helps prevent fumbling with the wrong plugs when trying to connect the cassette recorder — the red plug goes in the red jack, the gray plug goes in the smaller remote control jack and the black plug goes in the EARPHONE jack. Since the recorder needs to be plugged and unplugged several times during each use, this is a big timesaver.

While painting, you may also want to put a drop of paint on the edge of the volume and tone controls to show the correct setting for use with your Model 100. If the controls are white, the red nail polish will work fine, otherwise try a dab of typewriter opaquing fluid (see Figure 5). You'll find these marks to be a big help in setting the volume and tone controls, which also must be done with each use of the recorder.

Organizing Your Tapes

Two more items will make your tape library complete: a handful of three by five-inch cards and a tape cassette holder of some kind. The cassette tape storage unit isn't really necessary, but it makes your briefcase a lot neater if all the tapes are kept in one small box. Figure 2 shows two different types that are especially suitable for use in a briefcase.

The first is manufactured by Scotch (3M) as part of their "C-Box" Pushbutton Cassette Storage System (see Figure 6). The Scotch line of cassette accessories is usually carried by budget department stores and by electronic wholesalers. A similar product, though not quite as sturdy, is available from Long's Electronics. It's called the K-cet Modular Pop-Out Cassette Storage System (see Figure 7). It holds six cassettes and sells for $5.39 plus $1.59 for shipping. The catalog number is EMRK6H and Long's address is 2700 Crestwood Boulevard, Birmingham, AL 35210. Both systems have a separate slide-out box for each cassette that can interlock with adjacent boxes on top and bottom.

Now that you have all the raw materials, you're ready to start your library. Begin by labeling your tapes "1A" on side one and "1B" on side two of the first tape, "2A" on side one of the second tape, etc. Do this on tapes one through four. These will be your "dump" tapes. On the fifth tape, write "BASIC Library 1A and 1B." This tape will be set aside for keeping finished BASIC programs.

Now you'll want to prepare two three by five catalog cards for each one of the tapes. Follow the format given in Figure 8.

Use the front of the card (lined side only) so you have one card for each tape side. Make up a few extra cards to replace ones that get filled up later. Fill in the tape number and type (either DUMP or BASIC Library) now.

The way to use the dump tapes is to periodically dump all the files from the Model 100 to tape, thus backing up your entire system. How often this should be done depends on how much the computer is used, but a good starting point is to make a dump tape after

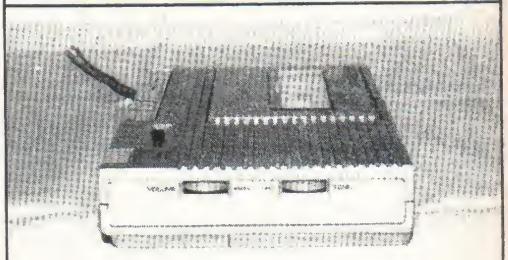
Figure 4

every five hours of work on the computer. If the system is used to write a lot of short letters, you might not want to take the trouble to dump them. Also, if there are files that seldom change, such as ADRS.DO, dump these after every change instead of every time a dump tape is made.

Saving A Text File

Let's use the example of ADRS.DO to see exactly what steps are necessary to make a dependable backup. Start with tape number 1A and rewind it to the beginning. If the tape has a leader, you'll need to advance the tape to the spot where it changes color. A ball-point pen exactly fits the cassette hub and allows you to pull the tape by turning the hub. Turn the right hub counter-clockwise until the intersection of the tape and leader shows in the middle opening of the front of the cassette (where the playback head normally goes). If you have a leaderless tape, this step can be skipped.

Figure 5



Now place the tape in the recorder and connect the cassette interface cable to the Model 100. Don't connect it to the recorder just yet. Adjust the volume and tone controls to the proper settings. If the recorder has a tape counter, press the reset button to set the counter to zero. Press the Record and Play buttons simultaneously, and speak into the

microphone as follows: "Dump of ADRS.DO made on mm/dd/yy" (indicate today's date). Press the Pause button to stop the recorder. When you say the filename, spell it out.

On line one of the tape catalog card for dump tape 1A, write in "ADRS.DO" under the NAME column and the tape counter reading under the START column. Connect the cassette interface cable to the recorder using the red and gray plugs. Press the Record and Play controls simultaneously. Place the cursor of the Model 100 on the file ADRS.DO and press ENTER. Press F3 (save) and key in ADRS and press ENTER. (You are not allowed to use a file extension such as .DO in tape filenames.)

Figure 6

When the tape stops, record the tape counter reading on line one of the catalog card under the column marked STOP. Fill in today's date under DATE. If you know the size of the file (in bytes) fill it in under SIZE. (See sidebar for a program to measure the size of a text file.) Press the stop button on the cassette recorder. Press F8 on the Model 100. Unplug all interface plugs from the cassette recorder.

Checking the Text File Dump

You now have a tentative copy of ADRS.DO on cassette. I say tentative because there may have been a hitch in the recording process. Sometimes a small particle (oxide) of tape comes off and sticks to the head of the recorder. This tiny speck can occasionally keep the tape from being readable if it is on the head when the recording is being made. Also, if it is a big flake of oxide, the spot on the tape where it came from never records successfully again. The only way to be sure of a good copy is to check it right after making it and be sure it is readable. Listing 1 shows a program that does this for you for .DO files. A checking program for BASIC programs is already built in to the Model 100.

If your tape copy turns out to be OK, place a check mark under the CK column on your catalog card. You are now ready to move on to the next file you want to back up. If your first file turns out to be defective, see the suggestions in "Trouble-shooting Guidelines." Press the Stop button on the recorder before disconnecting the interface cable.

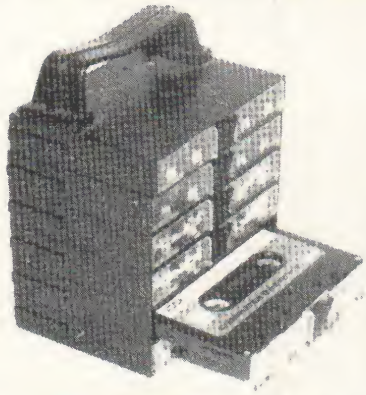
That's all there is to making a dependable copy of a file. If you want to dump additional files on the same tape at this time, just record a new voice header and then the data as before. Be sure to update the appropriate line on the catalog card for that side. When finished, remove the tape and place it in its drawer. You need not rewind the tape when you finish because next time it is used, you may want to use the other side, which would mean un-rewinding it before use.

Managing Your Tape Library

If you want to record a new file on a tape that already contains one or more files, here's what to do: As before, you'll need to reset the tape counter to zero at the beginning of the recordable part of the tape. Fast-forward the tape to the end of the last file on the tape you want to keep. This can be determined by comparing the value of the tape counter recorded under STOP on the last line of tape catalog card with the reading on the tape counter. Listen to a few seconds of the tape to ensure that you're actually past the end of the previous file. If you don't have a tape counter, listen to the voice headers to see where you are. Now you're ready to record a new voice header and dump the file just as described in the previous paragraph.

You should have no problem recording over an old file with new data if so desired, but there is one warning. Don't try to overwrite an existing file with a new one unless it's the last one on the tape that you want to keep. Here's an example: Say you have recorded files A, B and C (in that order) on a tape (see Figure 9). You no longer need File B, so you're ready to overwrite it with File D. If File D turns out to be longer than File B, it will also overwrite part of File C, making File C useless. The best way to avoid this problem is to never start a recording unless you're sure there's nothing later on the tape you want to keep.

Figure 7



That brings us to the subject of phasing out files. By the time you've used up all eight sides of the dump tapes (that's about 100 K), you will probably find that some of your oldest recordings are obsolete. For example, you may have several copies of ADRS.DO. Since

Sample Tape Library Catalog Card — **Figure 8**

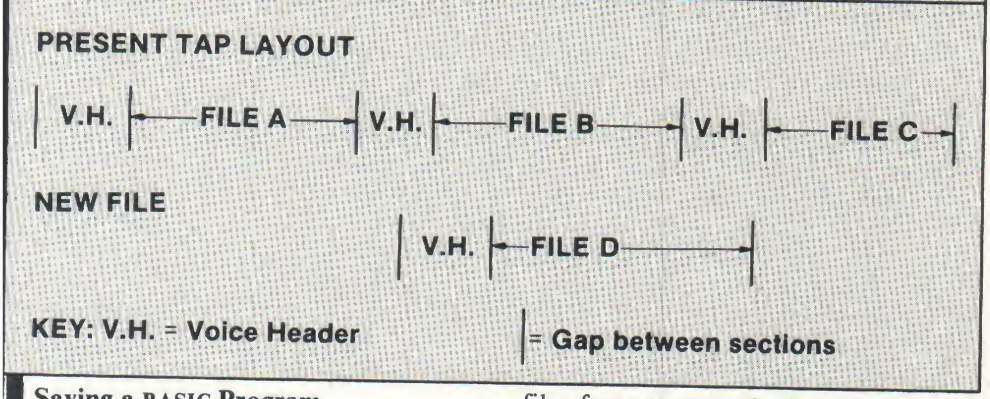
TAPE NUMBER _____ TYPE _____

#	NAME	START	STOP	SIZE	DATE	CK	NOTES
1							
2							
3							
4							
5							
6							
7							
8							
9							

you are probably not interested in keeping any but the most recent copy, the older copies are no longer needed.

When a file becomes obsolete, draw a line through it in the catalog card. If you strike out all of the lines on a card, then the corresponding tape side can be treated just like a new tape, and you can record over anything that's on it. If all but a couple of files on a side are obsolete, you may want to reload them and save them to a new tape, or simply make a hard copy instead. Anytime the last several files on a side are obsolete, you can record over them without disturbing the earlier files on the tape. If you ever get all eight sides filled up and can't find any files to delete, it's time to expand your library and buy additional tapes.

Figure 9 **Tape Layout Illustration**



Saving a BASIC Program

Making a tape copy of a BASIC program is just a little different. Let's use the example of the CHECK.BA file to see how. After finding an appropriate spot on a tape to record the new file, set up the recorder and make the voice header like this: "Dump of CHECK.BA dated mm/dd/yy." Plug in the red and gray plugs and press the Record and Play buttons simultaneously. As with a text file, place the cursor over the filename (CHECK.BA) and press ENTER. Immediately hold the SHIFT key down and hit the BREAK key. When you see the Ok prompt, type in CSAVE "CHECK and press ENTER. As usual, the .BA extension is not allowed for tape filenames. The closing double quote is optional. When the recorder stops, make a note of the tape counter reading.

In order to check the validity of a .BA file, use the built-in program *CLOAD?* like this. Rewind the tape to the voice header and stop it. Plug in the black and gray plugs and press the Play button. Then enter CLOAD? "CHECK and press ENTER. The Model 100 will reply Found: CHECK and proceed to compare it against the copy in memory. There is no further display until the file has been completely read, at which time the computer simply replies Ok, indicating the copy on tape is good. If you get this message instead, Verify failed, refer to the "Trouble-shooting Guidelines."

If the program being dumped to tape is one you're still refining, you'll want to use a dump tape. If it's a completely debugged program that you want to save for later use, copy it to your BASIC library tape. This library tape can be used to hold all of your completed programs as they are collected, and you should not need to phase out any of these files as with a dump tape.

Reloading Text Files

Now it's time to see how to reload files from tape to the Model 100. The first step is to find the filename listed on a catalog card. Compare the size listed on the catalog card against the available memory figure shown on the menu. If the file is larger than the space available, some files need to be deleted from the Model 100 first.

Load the appropriate tape in the recorder and reset the tape counter to zero at the beginning of the recordable part of the tape. Fast-forward the tape until the tape counter reads the same value as the number in the STOP column on the line *above* the entry for the tape you want. This puts you close to the beginning of the voice header for the file wanted. Press Play and listen to the voice header to verify it is the right file. As soon as you hear the date, press Pause. Connect the cassette interface cable to the Model 100 and plug in the black and gray plugs to the recorder. Release the Pause button.

If you're restoring a text file, place the cursor over the word TEXT and press ENTER. Upon seeing the prompt File to edit?, type in a filename of up to six characters. You may use the same name on the catalog card — it doesn't matter what this file is called. Then press ENTER and press F2. The Model 100 will prompt Load from:. Enter the name of the file as recorded on the catalog card, being sure to omit the extension .DO. When you press ENTER, the cassette tape begins moving and the file is loaded into the Model 100. When the tape stops, the file is shown on the LCD display, just as if you had typed it in yourself.

If you get the message Aborted, that means the Model 100 has had a problem in reading the file. If you followed the procedures outlined above in making and checking the file, you have nothing to fear if that message appears. You should still be able to recover the file. Just refer to the "Trouble-shooting Guidelines."

Reloading BASIC Files

If restoring a BASIC file, connect the recorder as just mentioned and press Play. Place the cursor over the word BASIC and press ENTER. When the Ok prompt appears, press F5 to list the contents of the BASIC workspace. If there is already a program here, SAVE it to RAM before reloading from cassette. Then enter CLOAD and press ENTER and wait while the program loads. The Model 100 will respond Found: filename when it starts loading, and then Ok when finished. Once again, if you get an error message refer to the "Trouble-shooting Guidelines." After loading the BASIC file from tape, you may want to save it under a separate filename. To do this, type in: SAVE "Program name and press ENTER.

Any six-letter filename may be used as a program name. In this case, the .BA extension is not required, nor is the closing double quote. After saving the program just loaded with the SAVE command, you'll probably want to enter NEW to erase the old copy of the program in the BASIC workspace to conserve memory.

Trouble-shooting Guidelines

If you have trouble reading a file, the problem can usually be overcome by checking just a few details. Start with the obvious. Are the volume and tone controls correctly set? Are the cables connected correctly? Are your batteries getting a little old? Your tapes may sound great to the human ear, but the Model 100 judges by different criteria. Try some new batteries or an AC adapter. If all these are OK, you probably have one of two problems:

First, the recording head may be dirty. This could happen even if you've recently cleaned the head because oxide can flake off of the tape at any time. The cure is simple, just follow the instructions for cleaning the head that came with the recorder or head-cleaning kit. Be sure to wait a minute before using the recorder again to give the head and rollers time to dry thoroughly. I find it easier to clean the head whenever I have a problem than to try to do it periodically.

After you've cleaned the head, try reading the file again. If it's OK, you're in business. If you've just recorded the tape and had trouble in checking it, try recording the file again. If the heads were dirty when the recording was made, cleaning the heads may have cured the problem.

If not, here's the second most common problem: There may be a bad spot on the tape that can prevent you from ever making a successful recording on that section of the tape. The bad spot may be so small it doesn't even effect the other side of the tape. If you are able to successfully record the same file on another tape and it checks out OK, then you've found the problem. On the catalog card for the bad tape, make a note of the problem on the line for the file you've just recorded. Next time you record on that tape, start at the end of the bad file and you'll probably have no trouble. You rarely see a C-10 tape with more than one bad spot on it.

By taking the trouble to check a file right after it is made you can protect yourself from both of these problems. Once you've verified that a file is readable from tape using one of the check programs, the chances are almost 100 percent you'll be able to load it later.

I hope you'll agree now that using cassettes really isn't too difficult if you're willing to take a few extra seconds to manage your cassette library thoughtfully. Good luck in your efforts to build a library that just suits your needs!

Program to Check a Text File Copy on Cassette

Listing 1 is designed to be used in verifying the integrity of text files once they've been copied on cassette. To use the program, enter it as shown under BASIC. Save it as a separate program by typing SAVE "CASCK and pressing ENTER, then type NEW and exit BASIC.

Listing 1:

PCM BAR CODED LISTING

```
10 CLS:CLEAR512:OPEN"CAS:"FORINPUTAS1
20 LINEINPUT#1,LN$:PRINTLN$
30 IFEOF(1)THENCLOSE:PRINT"-->File Ok<--":END
40 GOTO20
```

Once a .DO file is copied to cassette, you may verify that the copy is readable by rewinding the tape to the beginning of the file, connecting the cassette interface cable and pressing the Play button on the recorder. Then place the cursor over the filename CASCK.BA and press ENTER. The program will attempt to read and display the next file it finds on the tape. Each character is checked with the built-in checksum routine and displayed on the LCD screen, though the layout will not match the way it looks under the editor.

If an error is encountered, the program terminates with an IO Error message. Otherwise, the message --->File O.K.<--- appears on the screen at the end of the file. At this point you can KILL the text file with complete confidence that it can be restored later, provided you don't lose or damage the tape.

The only trick to using the program is that you must have at least 659 bytes of memory free (as shown on the menu). If you are saving a file because you have run out of memory, you need to get a little more memory free before this program can be run. Start by clearing the copy buffer by entering CLEAR 0,MAXRAM under BASIC. At this point you should have at least 192 bytes free. To make enough space for this program to run, 12 more lines of text need to be deleted. If you have any files you can afford to kill, this is an easy way out. If you have a backup of any files (such as ADRS.DO), you can kill it now and reload it from cassette later. If there isn't a file you can sacrifice, space can be made by deleting lines from a text file.

The best way to clear up space in a pinch is to plan for it in advance. If you make a file called JUNK.DO that consists of 12 lines of X's (40 x's per line), that file can be killed any time you run out of space and you'll have exactly enough space to run this program. This same technique can bail you out of a pinch if you run out of memory when using CUT and PASTE.

Listing 2:

PCM BAR CODED LISTING

```
10 CLS:CLEAR 2000:FILES
20 PRINT@204," ";:INPUT"File name";F$
30 OPEN F$ FOR INPUT AS 1
40 IF EOF(1)GOTO60
50 LINEINPUT #1,C$:C=C+1+LEN(C$):GOTO40
60 C=C+1:PRINT@226,"Size="C:CLOSE
```

Program to Compute the Size of a Text File

When saving files to cassette, it's always helpful to know how large they are, so when they're ready to be reloaded, you can be sure there is enough space in memory. Listing 2 counts the number of bytes contained in a .DO file.

To use the size program, first enter the program under BASIC and save it using the SAVE command as follows: SAVE "SIZE and press ENTER. To run the program, place the cursor over the filename SIZE.BA and press ENTER. You will see a list of all of the text files currently on your system. Select one file to check the size and enter its name (including the extension .DO). When the program has been run, make a note of the size of the file on the catalog card. PCM

Figure the financing that best suits your budget

The Loan Officer

By Leonard Hyre

Now let's see, if I buy that new station wagon at the car dealer's 7.9 percent rate and finance it for 36 months, can I swing it? What if I settle for the bank's 11.2 percent and spread it over 48 months? Does that fit my budget a bit better? If you find yourself pondering such thoughts, *The Loan Officer* may be just what you're looking for.

The Loan Officer was created with a couple of things in mind. The most obvious are amortization schedules, rate comparison based on different lengths of payments using the same rate and an ability to compare different interest rates against a total amount borrowed. At the same time, I have been wanting to show off a different way to create BASIC menus. I wrote the menu routine several months ago and hadn't come up with the right kind of program to use it with, until now. As a bonus, *The*

Leonard Hyre works as a claims representative for the Social Security Administration. He has written several articles for THE RAINBOW, PCM's sister publication for the Color Computer, and is the author of a number of commercial programs. He may be contacted at P.O. Box 403, Cambridge, MD 21613; 301-228-0064.

DISK POWER™

. . . AT LAST !!

*ULTRA*CHARGE YOUR TANDY PORTABLE DISK DRIVE WITH DISK POWER 100/200/102

- **A REAL DISK OPERATING SYSTEM**

All disk functions completely supported by **DISK POWER**. No need to rely on FLOPPY.CO or user-written programs for external utilities. **DISK POWER** is not just another disk manager program, it is the full featured DOS that the MODEL 100 should have had from the very beginning!

- **DISK ACCESSIBLE FROM BASIC & TEXT**

Data file I/O operations supported with familiar BASIC commands. Save and Load programs from within BASIC. From TEXT, every .DO file can be Loaded from or Saved to Disk by **DISK POWER**.

- **TELCOM ENHANCEMENTS**

Direct Access to Basic & TEXT from inside TELCOM. Kill, Load, Save or Edit files without losing telephone communications. You can see your files while uploading. With **DISK POWER**, it is all possible!

- **FULLY MENU DRIVEN**

Any DISK or RAM file is selected by the cursor. The stroke of a SINGLE function key will perform the desired task on the HIGHLIGHTED file. You may never have to remember or even type a filename again, when using **DISK POWER!**

- **NOT JUST "FOR HACKERS ONLY"**

Though programmers will be surprised by the versatility of **DISK POWER**, regular users will find our operating system FAST, easy to use AND easy to understand.

- **WRITTEN BY HUGO FERREYRA** (SoftPower)

In **DISK POWER**, you can expect to find the same ***ULTRA***-features, innovation & ease of use that all users & reviewers found in **TEXT POWER 100**, which made it **"THE BEST FORMATTER THERE IS FOR THE 100"**.

- **24 HR/DAY BBS TECHNICAL INFORMATION & SUPPORT HOTLINE: (514) 457-2917**

Use STAT **M8N1E**, and Log-on with ID: **DISK** using PASSWORD: **POWER**

DISK POWER 100/200/102 $49.95 US **TEXT POWER 100 $49.95** US

or SAVE by buying the ***ULTRA* POWER** program pack: DISK POWER/TEXT POWER 100 FOR ONLY **$89.95** US

To order *TEXT POWER 100*, specify printer(s) used. The basic price includes support for one printer. Please add $10.00 for each additional printer you wish supported. All U.S. orders shipped from within the USA. Absolutely no custom forms, sales tax or duty required. Add $3.00 S&H to each order.

No surcharge for VISA or MC
To order, call **TOLL FREE:**

USA (Except N.Y.)	**1-800-448-4511 Ext. 313**
NEW YORK STATE ONLY	**1-800-962-1480 Ext. 313**
ONTARIO & QUEBEC	**1-800-255-1133 Ext. 313**

or send CHECK or MONEY ORDER to:

ULTRA SOFT

P.O. Box 219, Ste Anne de Bellevue, Québec
CANADA H9X 3R9 (514) 457-9293 (9-5 EST)

Loan Officer can be compiled with the Microsoft BASIC Compiler. (Note: The new Quick BASIC Compiler does not work properly on the Tandy 1000.)

When the program is loaded and run, the user is provided with a simple menu offering several choices, including "Amortization," "Compare Length," "Compare Rates" and "Exit Program." The current selection is shown in inverse colors, and is changed by pressing the space bar. As the bar is pressed, the highlighted menu selection moves down one selection at a time, then back to the top choice. When the user decides on a selection, it is chosen by pressing the ENTER key while the selection is highlighted. Additionally, a "help window" displays what each selection can be expected to perform as it is highlighted on the screen. All in all, it makes an attractive and user-friendly way to display menu choices.

How it Works

The programming involved is not at all complicated. After the initial display is created, a control loop in lines 370 through 450 keeps track of what is going on. If the key pressed is the ENTER key, Line 390 forces the program out of the loop and into the routine requested. If the space bar is pressed, the selection indicator is advanced to the next menu choice by lines 400 through 440. As each choice is presented, a window displays what that particular choice will do.

The rest of the program is broken up into three main sections: the regular amortization process (lines 570-1140), comparison of interest rates (lines 1180-1530) and comparing different lengths of loan payment schedules (lines 1650-2000). A short EXIT routine is included in lines 1570 through 1620.

I have labeled strings and numerics descriptively in the listing to help you follow the program flow. The finance formulas are the common ones found in everyday math, dressed up for ease of program readability. Novice programmers might like to take note of the use of PRINT USING to format both the screen and the printed page when data is displayed. This under-used but powerful command can make life a lot easier for a BASIC programmer.

Making it Work

There is nothing tricky in typing in this program and it should run fine on any of the Tandy MS-DOS machines. If you do not wish to type it in, remember that PCM ON DISK is available. Also, Delphi Group MSDOS will have the program available for downloading (at a nominal charge). If you prefer, send me $6 to cover costs and I will send you the program on a disk. Address requests or questions to P.O. Box 403, Cambridge, MD 21613. Be sure to mention what model computer you own.

If you have the Microsoft BASIC Compiler (*not* Quick BASIC), this program is virtually ready to compile. You will need to change the EXIT routine to eliminate the "Exit to BASIC" choice. Add the following line: `125 DIM C$(4), LENG(4), RATE(4)`. This should take care of the changes needed to compile it.

I hope you find *The Loan Officer* a useful and entertaining program. I've enjoyed creating and sharing it with you. □

The listing:

```
1 '*****************************************
2 '*            The  LOAN OFFICER           *
3 '*                                        *
4 '*        (C) 2/86   Leonard Hyre         *
5 '*             Cambridge  MD              *
6 '*****************************************
7 '
8 'For the TANDY MSDOS series of computers
9 '
10Ø '******************
11Ø '* MENU  CREATION *
12Ø '******************
13Ø '
14Ø '
15Ø KEY OFF:SCREEN Ø,1:COLOR 7,4:CLS
16Ø HEADER$=CHR$(2Ø1)+STRING$(78,2Ø5)+CHR$(187)+CHR$(186)+SPACE$(24)+"T h e  L O
 A N   O F F I C E R"+SPACE$(24)+CHR$(186)+CHR$(2ØØ)+STRING$(78,2Ø5)+CHR$(188)
17Ø PRINT HEADER$
18Ø DEFDBL A,B:COUNT=Ø:PRIN=Ø:LENG=Ø:RATE=Ø:MONPAY=Ø
19Ø C$(1)=" Amortization   ":C$(2)=" Compare Length ":C$(3)=" Compare Rates  " :
C$(4)=" Exit Program   ":DW=1Ø
2ØØ TOP$=CHR$(2Ø1)+STRING$(16,CHR$(2Ø5))+CHR$(187)
21Ø BOT$=CHR$(2ØØ)+STRING$(16,CHR$(2Ø5))+CHR$(188)
22Ø LOCATE 7,3Ø:PRINT "   Loan  Menu   "
23Ø LOCATE 8,3Ø:PRINT "________________"
24Ø LOCATE 6,29:PRINT TOP$:LOCATE 14,29:PRINT BOT$:LOCATE 4,71:PRINT "by L.Hyre"
25Ø FOR SIDES=7 TO 13:LOCATE SIDES,29:PRINT CHR$(186):LOCATE SIDES,46:PRINT CHR$
(186):NEXT SIDES
26Ø LOCATE 1Ø,3Ø:COLOR Ø,3:PRINT C$(1):COLOR 6,Ø
```

PCMfest®

Chicago

May 23-25

RAINBOWfest

Your admission to PCMfest also entitles you to attend RAINBOWfest, the highly popular show for the Tandy Color Computer which will run concurrently with PCMfest at the same location.

You're invited to PCMfest, a great show that's just for your Tandy MS-DOS or portable computer. It's sponsored by PCM, *The Personal Computer Magazine for Tandy Computer Users*. See the great variety of products featured in the exhibits of PCM advertisers. Meet national experts on your computer and take part in a full lineup of free seminars on topics designed to help you the most.

The show opens Friday evening with a 7 p.m. to 10 p.m. session. It continues Saturday from 10 a.m. to 6 p.m. and Sunday from 11 a.m. to 4 p.m. Tickets may be obtained directly from PCM by mail or phone order. We'll also send you a special reservation form so you can take advantage of our special room rate at the Hyatt Regency-Woodfield ($60, single or double room). Come to PCMfest and let's celebrate the new generation of Tandy computers!

CoCo Community Breakfast

Steve Bjork

A top-flight 6809 programmer, Steve authored such programs as *Stellar Life* and *Ghana Bwana*. He is product developer for his own company, SRB Software.

Free Seminars

Jeff Parker
PCM Contributing Editor

Choosing an MS-DOS Machine to Fit Your Needs

Steve Bjork
President of SRB Software

User Interfaces

John Ross
Owner and Managing partner of Ross Computer Services

Concepts of the Data Communication

Ross Litton
Manager of Howard Medical

Communication and File Transfer between a 1000 and a Color Computer

Dick White
PCM Contributing Editor

Spreadsheets

John Gibney
Delphi National Sales Director

National Information Serivces vs. Local Bulletin Board Systems

Jim Reed and Danny Humphress
RAINBOW and PCM Managing Editors

Writing for Publication

YES, I'm coming to Chicago! I want to save by buying tickets now at the special advance sale price.

Please send me:

_____ three-day tickets at $9 each total _____

_____ one-day tickets at $7 each total _____

(Circle one) Friday Saturday Sunday

_____ Saturday Breakfast at $12 each total _____

Handling Charge $1 _____

TOTAL ENCLOSED _____

(U.S. Currency Only, Please)

☐ Also send me a hotel reservation card for the Hyatt Regency Woodfield ($60, single or double room).

Name _____

Address _____

City _____ State _____

Telephone _____ ZIP _____

Company _____

☐ Payment enclosed, OR ☐ Charge to:

☐ VISA ☐ MasterCard ☐ American Express

Account Number _____

Exp. Date _____

Signature _____

Make checks payable to: PCM. Mail to: PCMfest, The Falsoft Building, P.O. Box 385, U.S. Hwy. 42, Prospect, KY 40059. To make reservations by phone, call: (502) 228-4492.

Advance ticket deadline: May 16, 1986. Orders received less than two weeks prior to show opening will be held for you at the door. Tickets will also be available at the door at a slightly higher price. Children under 4, free; 4 and over, full admission price.

```
27Ø LOCATE 11,3Ø:PRINT C$(2)
28Ø LOCATE 12,3Ø:PRINT C$(3)
29Ø LOCATE 13,3Ø:PRINT C$(4)
3ØØ COLOR 3,Ø:LOCATE 16,28:PRINT"<SPACEBAR> to Change":LOCATE 17,28:PRINT"< ENTE
R >  to Select":COLOR 6,Ø
31Ø WIDE$=STRING$(4Ø,CHR$(2Ø5)):LOCATE 2Ø,19:PRINT CHR$(2Ø1);WIDE$;CHR$(187):LOC
ATE 23,19:PRINT CHR$(2ØØ);WIDE$;CHR$(188):SIDE$=CHR$(186)+STRING$(4Ø," ")+CHR$(1
86):FOR SIDES=21 TO 22:LOCATE SIDES,19:PRINT SIDE$:NEXT
32Ø LOCATE 21,2Ø:COLOR 6,Ø:PRINT" Amortization:  This option allows you":LOCATE
22,2Ø:PRINT" to see a loan listed month by month !"
33Ø LOCATE 1Ø,3Ø,Ø
34Ø '
35Ø '*** Choice Made Here ***
36Ø '
37Ø CH$=INKEY$:IF CH$="" THEN 37Ø
38Ø GOSUB 49Ø
39Ø IF ASC(CH$)=13 THEN 45Ø
4ØØ IF ASC(CH$)=32 THEN LOCATE DW,3Ø:COLOR 6,Ø:PRINT C$(DW-9):DW=DW+1
41Ø IF DW=14 THEN DW=DW-4
42Ø IF DW=1Ø THEN 5ØØ ELSE IF DW=11 THEN 51Ø ELSE IF DW=12 THEN 52Ø ELSE 53Ø
43Ø LOCATE DW,3Ø:COLOR Ø,3:PRINT C$(DW-9):SOUND (9Ø*DW),.1:COLOR 6,Ø
44Ø GOTO 37Ø
45Ø IF DW=1Ø THEN 57Ø ELSE IF DW=11 THEN 165Ø ELSE IF DW=12 THEN 118Ø ELSE IF DW
=13 THEN 157Ø
46Ø '
47Ø '***  HELP Windows ***
48Ø '
49Ø WIPE$=STRING$(4Ø," "):COLOR ,Ø:FOR WIPE=21 TO 22:LOCATE WIPE,2Ø:PRINT WIPE$:
NEXT:COLOR 6,Ø:RETURN
5ØØ LOCATE 21,2Ø:COLOR 6,Ø:PRINT" Amortization:  This option allows you":LOCATE
22,2Ø:PRINT" to see a loan listed month by month !":GOTO 43Ø
51Ø LOCATE 21,2Ø:COLOR 6,Ø:PRINT" Compare Length: What is the difference ":LOCAT
E 22,2Ø:PRINT" in financing longer or shorter periods.":GOTO 43Ø
52Ø LOCATE 21,2Ø:COLOR 6,Ø:PRINT" Compare Rates: This options allows you ":LOCAT
E 22,2Ø:PRINT" to compare available Interest  Rates ! ":GOTO 43Ø
53Ø LOCATE 21,2Ø:COLOR 6,Ø:PRINT" Exit PROGRAM- This selection lets you ":LOCATE
 22,2Ø:PRINT" exit 'LOAN OFFICER' to MSDOS or BASIC.":GOTO 43Ø
54Ø '
55Ø '*** AMORTIZATION PROGRAM ***
56Ø '
57Ø CLS:PRINT HEADER$
58Ø LINE INPUT " How Much Would You Like To Borrow?   $";PRIN$:PRIN=VAL(PRIN$)
59Ø LINE INPUT " How Many Months Will You Finance:     ";LENG$:LENG=VAL(LENG$)
6ØØ LINE INPUT " What Interest Rate Do You Anticipate? ";RATE$:RATE=VAL(RATE$)
61Ø A=RATE/12ØØ:B=1-1/(1+A)^LENG:LOAN=PRIN:MONPAY=PRIN*A/B
62Ø MONPAY=INT(MONPAY*1ØØ+.5)/1ØØ
63Ø COLOR 2:PRINT:PRINT STRING$(8Ø,"_"):PRINT"Monthly Payment Would Be $";:PRINT
 USING " ###,###.##";MONPAY:PRINT"The TOTAL Interest paid= $";:PRINT USING " ###
,###.##";(MONPAY*LENG)-PRIN
64Ø PRINT:PRINT STRING$(8Ø,"_"):COLOR 3
65Ø PRINT"SELECT ->     <S>creen Amort Lisc   <P>rinter Amort List   <M>enu"
66Ø AK$=INKEY$:IF AK$=""THEN 66Ø
67Ø IF AK$="S" OR AK$="s" THEN 71Ø
68Ø IF AK$="M" OR AK$="m" THEN RUN
69Ø IF AK$="P" OR AK$="p" THEN 91Ø
7ØØ GOTO 66Ø
71Ø GOSUB 87Ø
72Ø FOR ZZ=1 TO LENG
73Ø FIN$="
```

```
74Ø IF COUNT<12 GOTO 78Ø
75Ø PRINT "     ";FIN$:LOCATE 23,27:COLOR 7,4:PRINT "Press <ANY KEY> to Continue
"
76Ø AK$=INKEY$:IF AK$=""THEN 76Ø
77Ø COUNT=Ø:GOSUB 87Ø
78Ø IPAY=PRIN*A:IPAY=INT(IPAY*1ØØ+.5)/1ØØ
79Ø PPAY=MONPAY-IPAY:PRIN=PRIN-PPAY:PPAY=INT(PPAY*1ØØ+.5)/1ØØ:PRIN=INT(PRIN*1ØØ+
.5)/1ØØ
8ØØ PRINT "     °";:PRINT USING "###    ";ZZ;
81Ø PRINT"°";:PRINT USING "   ###,###.##";PRIN;:PRINT"°";:PRINT USING "  ###,###
.##";MONPAY;:PRINT "°";:PRINT USING "   ###,###.##";PPAY;:PRINT"°";:PRINT USING
"   ###,###.##";IPAY;:PRINT"  °"
82Ø COLOR 2:IF  PRIN<=Ø  THEN  84Ø
83Ø COUNT=COUNT+1:NEXT ZZ
84Ø PRINT "     ";FIN$:LOCATE 23,6:COLOR 7,4:PRINT"Hit <ANY KEY> to Continue..."
85Ø AK$=INKEY$:IF AK$=""THEN 85Ø
86Ø RUN
87Ø COLOR 2,Ø:CLS:COLOR 4,7:PRINT HEADER$:PRINT:COLOR 7,Ø:PRINT "Borrowing $";LO
AN;"FOR ";LENG;"MONTHS AT";RATE;"%"
88Ø COLOR 2,Ø:PRINT STRING$(8Ø,196)
89Ø PRINT "     PAYMENT═┬═══BALANCE═┬═══PAYMENT═┬═══PRINCIPAL═┬═══INTEREST
ë¡
9ØØ RETURN
91Ø COLOR 3,1:PRINT:PRINT" Printer selected.......Advance paper to proper locati
on and <PRESS> any key. ":COLOR 6,Ø
92Ø AK$=INKEY$:IF AK$=""THEN 92Ø
93Ø PRINT:PRINT" Please Wait.....Printer Operation in Progress."
94Ø LPRINT STRING$(8Ø,"_"):LPRINT:LPRINT SPACE$(24);:LPRINT"T H E  L O A N  O F
 F I C E R":LPRINT STRING$(8Ø,"_"):LPRINT
95Ø LPRINT"AMOUNT OF LOAN: $":LOAN:" at ":RATE;"% For ";LENG;" Monthes."
96Ø LPRINT STRING$(8Ø,"_"):LPRINT"PAYMENT #      BALANCE      PAYMENT    PRINCIP
AL     INTEREST"
97Ø FOR ZZ=1 TO LENG
98Ø IF COUNT<48 THEN 1Ø4Ø :REM PRINT 48 LINES PER PAGE
99Ø FOR NXTPG=1 TO 1Ø:LPRINT:NEXT
1ØØØ LPRINT STRING$(8Ø,"_"):LPRINT:LPRINT SPACE$(24);:LPRINT"T H E  L O A N  O
F F I C E R":LPRINT STRING$(8Ø,"_"):LPRINT
1Ø1Ø LPRINT"AMOUNT OF LOAN: $";LOAN;" at ";RATE;"% For ";LENG;" Monthes."
1Ø2Ø LPRINT STRING$(8Ø,"_"):LPRINT"PAYMENT #      BALANCE      PAYMENT    PRINCI
PAL     INTEREST"
1Ø3Ø COUNT=Ø
1Ø4Ø IPAY=PRIN*A:IPAY=INT(IPAY*1ØØ+.5)/1ØØ
1Ø5Ø PPAY=MONPAY-IPAY:PRIN=PRIN-PPAY:PPAY=INT(PPAY*1ØØ+.5)/1ØØ:PRIN=INT(PRIN*1ØØ
+.5)/1ØØ
1Ø6Ø LPRINT USING "####     ";ZZ;
1Ø7Ø LPRINT USING "   ###,###.##";PRIN;MONPAY;PPAY;IPAY
1Ø8Ø IF PRIN <=Ø THEN 11ØØ
1Ø9Ø COUNT=COUNT+1:NEXT ZZ
11ØØ LPRINT STRING$(8Ø,"_")
111Ø PRINT:PRINT" PRINTING OPERATON COMPLETED!"
112Ø COLOR 7,4:PRINT:PRINT" Hit <ANY KEY> to Continue...."
113Ø AK$=INKEY$:IF AK$=""THEN 113Ø
114Ø RUN
115Ø '
116Ø '*** COMPARING INTEREST RATES ***
117Ø '
118Ø CLS:PRINT HEADER$;:COLOR 2:LOCATE CSRLIN,22:PRINT"Comparison of Different I
nterest Rates":PRINT STRING$(8Ø,"_"):PRINT
119Ø '
```

```
12ØØ COLOR 7:INPUT"Enter the Amount of the Loan";PRIN:INPUT"How Long Will You Fi
nance (in Months) ";LENG:RT=1:COLOR 2:PRINT STRING$(8Ø,"_")
121Ø PRINT"Enter RATES to compare (Up to 5 may be entered or enter 'Ø' to end) ?
122Ø FOR X=1 TO 5
123Ø PRINT "Interest Rate #";X;:PRINT"=>";:INPUT RATE(RT)
124Ø IF RATE(RT)=Ø THEN 127Ø
125Ø RT=RT+1
126Ø NEXT X
127Ø FOR WIPE=6 TO 2Ø:LOCATE WIPE,1:PRINT SPACE$(8Ø):NEXT WIPE:LOCATE 6,1
128Ø PRINT "Length of Loan is ";:PRINT USING "###";LENG;:PRINT " Months."
129Ø COLOR 7
13ØØ PRINT CHR$(213);STRING$(78,2Ø5);CHR$(184);
131Ø FOR Y=1 TO 5
132Ø IF RATE(Y)=Ø THEN 14ØØ
133Ø A=RATE(Y)/12ØØ:B=1-1/(1+A)^LENG:MONPAY=PRIN*A/B
134Ø MONPAY=INT(MONPAY*1ØØ+.5)/1ØØ:COST=MONPAY*LENG
135Ø COLOR 7:PRINT CHR$(179);"  Borrow $";:PRINT USING "###,###.##";PRIN;:PRINT"
 at";:PRINT USING " ##.##";RATE(Y);:PRINT " %";:COLOR 6:PRINT "  PAYMENTS= $";:P
RINT USING "##,###.##";MONPAY;:COLOR 2:PRINT"  INTEREST=$";:PRINT USING "###,###
.##";COST-PRIN;

136Ø COLOR 7:LOCATE CSRLIN,8Ø:PRINT CHR$(179);
137Ø PRINT CHR$(212);STRING$(78,2Ø5);CHR$(19Ø);
138Ø PRINT CHR$(213);STRING$(78,2Ø5);CHR$(184);
139Ø NEXT
14ØØ PRINT CHR$(179);:COLOR 4:PRINT "            Press <P> for PRINTER - Else An
y other Key For MENU.";:LOCATE CSRLIN,8Ø:COLOR 7:PRINT CHR$(179);
141Ø PRINT CHR$(212);STRING$(78,2Ø5);CHR$(19Ø);
142Ø AK$=INKEY$:IF AK$=""THEN 142Ø
143Ø IF AK$="P" OR AK$="p"THEN 144Ø ELSE RUN
144Ø LPRINT STRING$(8Ø,"_"):LPRINT:LPRINT SPACE$(24);"T H E  L O A N  O F F I C
E R":LPRINT STRING$(8Ø,"_"):LPRINT
145Ø LPRINT"  LENGTH OF LOAN IS ";:LPRINT USING "###";LENG;:LPRINT" months."
146Ø LPRINT STRING$(8Ø,"_")
147Ø FOR Z= 1 TO 5
148Ø IF RATE(Z)=Ø THEN 153Ø
149Ø A=RATE(Z)/12ØØ:B=1-1/(1+A)^LENG:MONPAY=PRIN*A/B
15ØØ MONPAY=INT(MONPAY*1ØØ+.5)/1ØØ:COST=MONPAY*LENG
151Ø LPRINT "  Borrow $";:LPRINT USING "###,###.##";PRIN;:LPRINT" at";:LPRINT US
ING " ##.##";RATE(Z);:LPRINT " %";:LPRINT "  PAYMENTS= $";:LPRINT USING "##,###.
##";MONPAY;:LPRINT"  INTEREST= $";:LPRINT USING "###,###.##";COST-PRIN
152Ø LPRINT STRING$(8Ø,"_"):NEXT Z
153Ø RUN
154Ø '
155Ø ' *** EXIT OPTIONS ***
156Ø '
157Ø CLS:PRINT HEADER$:PRINT:PRINT
158Ø PRINT"Please Choose From Menu:":PRINT"          1> Exit to BASIC":PRINT"
       2> Exit to MSDOS":PRINT"          3> Do NOT Exit!"
159Ø AK$=INKEY$:IF AK$=""THEN 159Ø
16ØØ IF AK$="1"THEN CLS:NEW
161Ø IF AK$="2"THEN SYSTEM
162Ø RUN
163Ø ' *** COMPARE DIFFERENT LENGTH LOAN PERIODS ***
164Ø '
165Ø CLS:PRINT HEADER$;:PRINT"             Comparison of finance costs for diff
erent time periods.":COLOR 2
166Ø INPUT"Amount of LOAN";PRIN:INPUT"INTEREST RATE";RATE
167Ø PRINT"  Enter TIME PERIODS to compare (Up to 5 may be entered or enter 'Ø'
```

```
to end) ?":COLOR 3:PRINT"  [ MUST BE AS TOTAL NO. OF MONTHS ]":COLOR 2:RT=1
168Ø FOR X=1 TO 5
169Ø PRINT"Period No.";X;:INPUT LENG(RT)
17ØØ IF LENG(RT)=Ø THEN 173Ø
171Ø RT=RT+1
172Ø NEXT X
173Ø FOR WIPE=5 TO 2Ø:LOCATE WIPE,1:PRINT SPACE$(8Ø):NEXT WIPE:LOCATE 6,1
174Ø PRINT"   For A LOAN of $";:PRINT USING" ###,###.##";PRIN;:PRINT" at ";:PRIN
T USING " ##.##";RATE;:PRINT "% Interest..."
175Ø COLOR 7
176Ø PRINT CHR$(213);STRING$(78,2Ø5);CHR$(184);
177Ø FOR Y=1 TO 5
178Ø IF LENG(Y)=Ø THEN 186Ø
179Ø A=RATE/12ØØ:B=1-1/(1+A)^LENG(Y):MONPAY=PRIN*A/B
18ØØ MONPAY=INT(MONPAY*1ØØ+.5)/1ØØ:COST=MONPAY*LENG(Y)
181Ø PRINT CHR$(179);"  Length-";:PRINT USING "###";LENG(Y);:PRINT" Months";:COL
OR 6:PRINT"   PAYMENTS at $";:PRINT USING "##,###.##";MONPAY;:PRINT" /Mo.";:COLO
R 2:PRINT"   TOTAL INT= $";:PRINT USING "###,###.##";COST-PRIN;
182Ø COLOR 7:LOCATE CSRLIN,8Ø:PRINT CHR$(179);
183Ø PRINT CHR$(212);STRING$(78,2Ø5);CHR$(19Ø);
184Ø PRINT CHR$(213);STRING$(78,2Ø5);CHR$(184);
185Ø NEXT Y
186Ø PRINT CHR$(179);:COLOR 4:PRINT"                    Press <P> for PRINTER - Else
Any Other Key For MENU.";:LOCATE CSRLIN,8Ø:COLOR 7:PRINT CHR$(179);
187Ø PRINT CHR$(212);STRING$(78,2Ø5);CHR$(19Ø);
188Ø AK$=INKEY$:IF AK$=""THEN 188Ø
189Ø IF AK$="P" OR AK$="p"THEN 19ØØ ELSE RUN
19ØØ LPRINT STRING$(8Ø,"_"):LPRINT:LPRINT SPACE$(24);"T H E  L O A N  O F F I C
E R":LPRINT STRING$(8Ø,"_"):LPRINT
191Ø LPRINT"  For A LOAN of $";:LPRINT USING" ###,###.##";PRIN;:LPRINT" at ";:LP
RINT USING " ##.##";RATE;:LPRINT "% Interest..."
192Ø LPRINT STRING$(8Ø," ")
193Ø FOR Z= 1 TO 5
194Ø IF LENG(Z)=Ø THEN 2ØØØ
195Ø A=RATE/12ØØ:B=1-1/(1+A)^LENG(Z):MONPAY=PRIN*A/B
196Ø MONPAY=INT(MONPAY*1ØØ+.5)/1ØØ:COST=MONPAY*LENG(Z)
197Ø LPRINT "  Length-";:LPRINT USING "###";LENG(Z);:LPRINT" Months   PAYMENTS a
t $";:LPRINT USING "##,###.##";MONPAY;:LPRINT" /Mo.  TOTAL INT= $";:LPRINT USING
 "###,###.##";COST-PRIN
198Ø LPRINT STRING$(8Ø,"_")
199Ø NEXT Z
2ØØØ RUN
```

PCM

TANDY IS DANDY...

...until you compare

	Tandy Memory Plus	Zuckerboard Expansion Memory
DMA	Inc.	**Inc.**
with 256K	$319.95	**$109**
with 512K	$519.90	**$149**
Options:		
Clock/Calendar	$ 99.95*	**$49**
256K Upgrade	$199.95	**$49**
RAM Disk/ Print Spooler	N/A	**$49**
Warranty	90 Days *with mouse	**2 Years**

MADE IN U.S.A.

INTRODUCTORY OFFER PCMA

*Send the **Zuckerboards** I need for my TANDY 1000 right away.*
Here's my check ☐ or money order ☐ **MADE PAYABLE TO A.T.D.**

____EM with 256K	@ $109 = ____	Subtotal	____
____EM with 512K	@ $149 = ____	CA residents	
____Clock/calendar	@ $ 49 = ____	add 7% tax	____
____256K upgrade	@ $ 49 = ____	Shipping	$6.00
____RAM Disk/ Print spooler	@ $ 49 = ____	**Amount Enclosed**	____
____Multifunction Board	@ $249 = ____		
____10 MB Hard Disk	@ $529 = ____		
____20 MB Hard Disk	@ $599 = ____		

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

TELEPHONE ( ) _____

A.T.D. 1287 Lawrence Station Rd., Sunnyvale, CA 94089

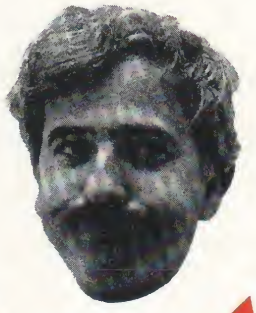
NEW!!

ZUCKERBOARD PRODUCTS FOR THE TANDY 1000

Multifunction Board **$249**
Includes DMA controller chip, 512K memory, Clock/calendar with 20 year battery, Serial port, and Ramdisk/Print spooler software.

Internal Hard Disk **10 Mbyte** **$529**
20 Mbyte **$599**
Half-height. Includes controller and cables. Mounts internally, above floppy drive A. Requires system with DMA already installed.

CALL TOLL-FREE TO ORDER
CA (800) 233-6874
(800) 222-4920



Dr. Dr. Zucker

It's another ZUCKERBOARD

ATD 1287 Lawrence Station Road • Sunnyvale, CA 94089 • (408) 734-4631

All prices subject to change without notice due to fluctuations in the chip market.

Custom design your own video character fonts

Video Character Editor

By Davy Crockett

While perusing the Memory Map for the Tandy 2000 I noticed a very interesting entry: "F800:0 to F9FF:0 Video Character RAM." The key word here is "RAM." This means you can write information into it instead of only being able to read from it. Well, my eyes lit up! You mean I can actually change the characters that display on the screen? A small experiment confirmed that it can be done. Enter the short program contained in Figure 1 and run it. If you enter everything correctly, the entire screen will be lined just like notebook paper. Change the 255 in Line 30 to zero and run it again to remove the lines.

The Tandy 2000 uses a block of pixels, eight across by 16 down, to display each character on the screen. Each pixel corresponds to a specific bit in the video character RAM. It takes 16 bytes with eight bits each to define a character for the video screen. These 16 bytes are interspersed with what seems to be attribute bytes, but the attribute bytes are ignored by the video display hardware. Therefore, each character requires 32 bytes to completely define its pattern of on and off pixels. The

character 'A' is stored in the sixty-fifth block of 32 bytes with the pattern for each raster scan line in the even-numbered bytes as shown in Figure 2.

Publishers use the term "font" to refer to the complete collection of letters and symbols of the same size and style. I will use the same term to apply to the collection of pixel patterns that define characters which may be displayed on the video screen. The program EDIT FONT.BAS allows you to edit these pixel patterns and change the characters.

The original (or default) font is initialized in the video character RAM during the boot-up sequence. EDITFONT refers to this area in memory as the "active font" since any changes to the values immediately affect what is displayed on the screen. In addition to the active font, EDITFONT uses the upper portion of BASIC's work area to store a "working font." Changes made to the working font do not take effect until the working font is moved to the active font memory locations. Space for the working font must be reserved by using the "/M" switch with a value of 57343 when loading BASIC. I use a batch file containing the single line: BASIC EDITFONT /M:57343.

Editing a Font

Initially the work area contains random garbage left over from whatever program happened to execute before EDITFONT. To begin an edit session you must first load the work area with a valid font. The work area may be loaded from the active area or from a font that you have previously saved to disk. The first time the program is used you will have no fonts on disk, so the easiest way to begin is to transfer the active font to the working font using the F6 key. That way you begin with a complete set of characters that only need modification.

Once a font is loaded you may select a character to modify. Selection is accomplished by simply pressing the key for the character you wish to modify. When you wish to modify a character that is not found on the keyboard, hold down the ALT key and enter the decimal number corresponding to the character on the numeric keypad.

Once a character has been selected you are placed in edit mode. The bit pattern for that character is displayed in four areas on the screen. Two blocks contain an oversized representation of the pixel pattern. An asterisk is used to represent a pixel being turned on and a blank means turned off. The second two smaller blocks show the character as it will actually be seen in double-width mode on the screen. The two blocks on the right remain constant until you exit edit mode, while the two blocks on the left change immediately when you set/reset pixels.

Bit patterns may be changed by moving the cursor around in the large block and setting or re-setting the pixels. Pressing the F1 key sets the current pixel to on; F2 re-sets the pixel (turns it off). When you are satisfied with the results, save the character back to the working font by pressing the F3 key. If at any time you become displeased with the results

Figure 1: Video Character RAM Demonstration

```
10 DEF SEG = &HF800
20 FOR CHAR = 0 TO 255
30    POKE CHAR * 32 + 26, 255
40 NEXT CHAR
50 DEF SEG : STOP
```

Figure 2: Bit pattern and memory locations for the capital 'A' character (41 Hex, 65 decimal)

```
      Bits              Bits                Bits                    Address
      76543210          76543210            76543210                SEG=F800H
    +---------+       +---------+           Binary        Hex       Offset
  0 |         |       | byte  0 |  =  00000000B  =  00H  :  65 * 32 +  0
  2 |    *    |       | byte  2 |  =  00001000B  =  08H  :  65 * 32 +  2
  4 |   * *   |       | byte  4 |  =  00010100B  =  14H  :  65 * 32 +  4
  6 |  *   *  |       | byte  6 |  =  00100010B  =  22H  :  65 * 32 +  6
  8 | *     * |       | byte  8 |  =  01000001B  =  41H  :  65 * 32 +  8
B 10| *     * |       | byte 10 |  =  01000001B  =  41H  :  65 * 32 + 10
Y 12| ******* |       | byte 12 |  =  01111111B  =  7FH  :  65 * 32 + 12
T 14| *     * |       | byte 14 |  =  01000001B  =  41H  :  65 * 32 + 14
E 16| *     * |       | byte 16 |  =  01000001B  =  41H  :  65 * 32 + 16
S 18| *     * |       | byte 18 |  =  01000001B  =  41H  :  65 * 32 + 18
  20|         |       | byte 20 |  =  00000000B  =  00H  :  65 * 32 + 20
  22|         |       | byte 22 |  =  00000000B  =  00H  :  65 * 32 + 22
  24|         |       | byte 24 |  =  00000000B  =  00H  :  65 * 32 + 24
  26|         |       | byte 26 |  =  00000000B  =  00H  :  65 * 32 + 26
  28|         |       | byte 28 |  =  00000000B  =  00H  :  65 * 32 + 28
  30|         |       | byte 30 |  =  00000000B  =  00H  :  65 * 32 + 30
    +---------+       +---------+
```

of your editing, press the F4 key and any changes you have made are ignored. Don't worry if you cannot remember these 'F' key functions — a little reference guide similar to Figure 3 is always on the screen.

Disk File Fonts

EDITFONT.BAS will not allow you to exit by pressing the F9 key until you first save the font you have been working on to disk. This protects you in case the F9 key is accidentally pressed.

Pressing the F12 key saves the working font to a disk file. You will be prompted for the font name. Do not include an extension, as EDITFONT.BAS automatically adds .FNT for a filename extension. In addition to saving the working font to a disk file, you may simply transfer the working font to the active font. Function key F10 can then display the complete set of ASCII characters generated so you can see what they look like together.

The last two options on the functions menu allow you to save and load the active font directly from a disk file. The default extension of .FNT is supplied for these files, so you should not enter an extension.

Loading the Active Font

Now that you have a font in a disk file, how do you get it loaded into the video character RAM? You could write a short BASIC program that executes in an AUTOEXEC.BAT file and utilizes the BLOAD command to load the font. I found this technique slow and, due to the inefficient batch processing by MS-DOS, it generated a very unpleasant thrashing sound in the disk drive. I figured there must be a better way. Thus was born LOAD FONT.COM, shown in the second program listing. It accepts an .FNT file through the standard input and loads it to the correct memory locations. You can include the command LOADFONT<fontname.FNT in your AUTOEXEC.BAT file to load a font automatically when booting up. For those who do not have an assembler, I have included a BASIC program that generates the LOADFONT.COM program.

I have used EDITFONT.BAS to create a font containing all of the Model I/III graphics characters and to run programs I have converted from the Model III. Instead of changing all of the CHR$(xx) statements, I simply include the statement: DEF SEG = &HF800 : BLOAD "GRAPHICS.FNT", 0 : DEF SEG.

Figure 3: EDITFONT.BAS **Function Key Definitions**

Edit Mode Function Keys

F1	Set a pixel (turn it on)
F2	Reset a pixel, (turn it off)
F3	End edit mode and save the character pattern
F4	End edit mode without saving the character

Select Mode Function Keys

F5	Transfer working font to active font
F6	Transfer active font to working font
F7	Load the active font from a disk file
F8	Save the active font to a disk file
F9	Exit the program to MS-DOS
F10	Display characters using the active font
F11	Load the working font from a disk file
F12	Save the working font to a disk file

Listing 1:

```
100 '*=============================================================*
110 '*                                                             *
120 '*     ****************************        EDITFONT.BAS      *
130 '*     *****  EDITFONT.BAS  *****         Version 1.0       *
140 '*     ****************************        13 April 1985     *
150 '*                                                             *
160 '*     Davy Crockett                                           *
170 '*     5807 Cherrywood Lane                                    *
180 '*     Apartment #104                                          *
190 '*     Greenbelt, Maryland   20770                             *
200 '*                                                             *
210 '*     This program will allow you to edit the character       *
220 '*     bit patterns for the Tandy 2000 Monochrome Video        *
230 '*     display.  These bit patterns are stored in memory       *
240 '*     at segment &HF800 in the address space.                 *
250 '*                                                             *
260 '*     Execute with command:  basic editfont /M:57343          *
270 '*                                                             *
280 '*=============================================================*
290 '
300 '       Initialization Section.
310 '
320 DEFINT A-Z : WIDTH 40 : KEY OFF : CLS : GOSUB 1790 : GOTO 1630
330 '
340 '     Set a bit routine (F1)
```

TURBO-M™

Check Out Our Systems — High **QUALITY 4** layer ***TURBO*** motherboard at the **LOWEST** prices.
Our hard drive systems — an **EXTRA** megabyte of Storage ***FREE***
(more megabytes for less megabucks)
Our monitors are high resolutions with minimum 1,000 lines
— They even have their own tilt and swivel base ***at no added cost.*** —
Powered by a heavy duty power supply, and offering ***TRUE*** turbo speed.
ONE FULL YEAR WARRANTY

COMPARE OUR SYSTEMS — TURBO M is the COMPETITION

SYSTEM-A-SINGLE FLOPPY SYSTEM
$899.95!

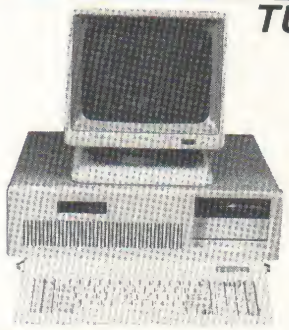
SYSTEM 22-HARD DRIVE SYSTEM
22 MEGABYTE SYSTEM
$1,399.95!

TURBO M PC COMPLETE SYSTEM
Dual Speed CPU — 4.77 AND TURBO
NEC V20 PROCESSOR

- 640 K memory
- 8 slot motherboard
- 135 Watt Power Supply
- Monochrome TTL Card
- Monochrome Display High Res
- Tilt and Swivel Base Monitor
- Serial port (2nd port opt.)
- 2 parallel ports
- Clock calendar
- Game Port
- Keyboard 5151 Style
- Ram spooler and
- Printer spooler system
- DOS 2.11

Toll Free Ordering 1-800-343-8841

TURBO M / AT—COMPLETE SYSTEMS
STARTING AT $2,359.95 !!!
ONE FULL YEAR WARRANTY



Comes Complete with

- 6 and 8 MHZ Switch Selectable
- 8 Slot Motherboard
- 192 Watt Power Supply
- 1.2 Megabyte Floppy Drive
- 1 Full Megabyte of Memory on Motherboard
- 2 Parallel Ports — 2 Serial Ports
- Clock Calendar
- Hercules Compatible Monochrome Card
- Monochrome Display-High Resolution
- Tilt and Swivel Base
- DOS 3.1 — AT Keyboard

ADD-IN BOARDS TO HELP YOU BUILD YOUR SYSTEM

TURBO-M Motherboad Dual Speed **$299.95**	Monochrome Graphics Card with Printer Port **$94.95**	Floppy I/O Card Controls Two Floppy Drives. Serial Port Parallel Port Clock Calendar Game Port **$109.95**	Winchester Hard Drive Controller **$139.95**	10 Megabyte and 20 Megabyte Drives Complete System Drive, Controller, Cables and easy to follow instructions. **$369.95 — $469.95**		Color Graphics Card with Printer Port **$118.95**
135 Watt Power Supply **$89.95**	Multi Function Card with Space for 384K of Memory. Serial Port. Parallel Port Clock Calendar. Game Port **$89.95**	5151 Compatible Keyboard with Separate Cursor and Number Pads **$124.95**	Monochrome Monitor with High Resolution Tilt/Swivel Base **$179.95**	Floppy Drive Card Control up to Four Drives **$79.95**	P.C DOS 3.1 **$80.00**	Flip Top Cabinet with Speaker **$64.95**

MEGADISK™ HARD DRIVES

Our Hard Drive Systems Are Compatible With
IBM PC and AT, HEATH/ZENITH, COMPAQ, COLUMBIA, PC CLONES, TAVA, EAGLE
TRS/80 Models I / III / IV / 4D / 4P / MAX 80 / COLOR COMPUTER, Tandy 1000

Systems come complete with drive, controller, cables, hardware and easy to follow instructions.

IBM Etc.
- 5 megabytes ..... starts at $239.95
- 10 megabytes ..... starts at $369.95
- 20 megabytes ..... starts at $469.95
- 60 megabytes ..... starts at $1,299.95

Systems come complete with one free software driver for TRSDOS 6, LDOS 5.1.x, DOSPLUS 3.4/4, or NEWDOS 80.CP/M is available.

TRS/80
- 5 megabytes $449.95
- 10 megabytes $679.95
- 15 megabytes $749.95
- 20 megabytes $849.95
- 24 megabytes $899.95
- 30 megabyte-fixed & removeable $1,399.95
- 40 megabytes $1,299.95
- 50 megabytes $1,449.95

REMOVABLE MEDIA SPECIAL
5 Megabyte Cartridge Hard Drive Now Specially Priced!

Warranty Information: TURBO-M systems come with a full year warranty for parts and labor.
TURBO-M systems are guaranteed to run all programs such as LOTUS, SYMPHONY, AUTOCAD, FLIGHT SIMULATOR, DBASE, MICROGRAPFX, WORDSTAR, LEADING EDGE W/P, RBASE.
MEGADISK hard drives are fully warranteed for parts and labor.

TERMS and CONDITIONS:
All prices are cash discounted. However, we do accept MC, VISA, AMEX & DISCOVER credit cards. Please inquire.
C.O.D.'s are accepted-No deposit required.
Purchase Orders-Corporate, Government & School P.O.'s are accepted. Please call for details.
Shipping Costs are calculated per order. Please call for total.
Shipments of all in-stock products are made within 24 hours, same day service is available upon request at no added cost.
Not responsible for typographical errors.

SOFTWARE SUPPORT, INC
1 Edgell Road
Framingham, MA 01701
1-617-872-9090
Hours: Mon.-Fri. 10 am to 5:30 pm (est) Sat 4:00 pm
Toll Free Ordering 1-800-343-8841
Dealer Inquiries Invited

Service & Returns: It is our policy to repair all service returns within 24-48 hours. Normally same day turn-a-round is accomplished. It is necessary to have a (Rieturn (M)aterial (A)uthorization to insure speedy service.

```
35Ø '
36Ø IF NOT ED THEN RETURN ELSE PRINT BS$; : N(Y)=N(Y) OR 2^X : GOTO 41Ø
37Ø '
38Ø '    Reset a bit routine (F2)
39Ø '
4ØØ IF NOT ED THEN RETURN ELSE PRINT BR$; : N(Y)=(N(Y) OR 2^X) XOR 2^X
41Ø DEF SEG = ACTIVE.SEG : POKE 32+Y*2,N(Y) : DEF SEG : GOTO 75Ø
42Ø '
43Ø '    End Edit and save character (F3)
44Ø '
45Ø IF NOT ED THEN RETURN ELSE ED=F% : SV=T%
46Ø FOR K=Ø TO 15 : POKE WORK.OFF+BYTE%*32+K*2,N(K) : NEXT : RETURN
47Ø '
48Ø '    End Edit and do not save character (F4)
49Ø '
5ØØ IF NOT ED THEN RETURN ELSE ED=F% : RETURN
51Ø '
52Ø '    Working Font to Active Font (F5)
53Ø '
54Ø GOSUB 154Ø : LOCATE 24,2 : PRINT "* Loading Active Font";
55Ø FOR M=3 TO 255 : FOR K=Ø TO 15 : DEF SEG : B=PEEK(WORK.OFF+M*32+K*2)
56Ø DEF SEG = ACTIVE.SEG : POKE M*32+K*2,B : NEXT : NEXT : GOSUB 154Ø : RETURN
57Ø '
58Ø '    Active Font to Working Font (F6)
59Ø '
6ØØ GOSUB 154Ø : LOCATE 24,2 : PRINT "* Loading Working Font";
61Ø FOR M=3 TO 255 : FOR K=Ø TO 15 : DEF SEG = ACTIVE.SEG : B=PEEK(M*32+K*2)
62Ø DEF SEG : POKE WORK.OFF+M*32+K*2,B : NEXT : NEXT : GOSUB 154Ø : RETURN
63Ø '
64Ø '    Return to MS-DOS (F9)
65Ø '
66Ø IF NOT SV THEN LOCATE 23,2 : PRINT ">>==> Nothing Saved!"; : SV=T% : RETURN
67Ø WIDTH 8Ø : CLS : SYSTEM
68Ø '
69Ø '    Cursor control routines
7ØØ '
71Ø X=7 : Y=Ø       : LOCATE Y+2,1Ø-X,1 : RETURN :'initial location
72Ø Y=(Y-1) AND 15 : LOCATE Y+2,1Ø-X,1 : RETURN :'up arrow
73Ø Y=(Y+1) AND 15 : LOCATE Y+2,1Ø-X,1 : RETURN :'down arrow
74Ø X=(X+1) AND  7 : LOCATE Y+2,1Ø-X,1 : RETURN :'left arrow
75Ø X=(X-1) AND  7 : LOCATE Y+2,1Ø-X,1 : RETURN :'right arrow
76Ø '
77Ø '    Disk Font to Working Font (F11)
78Ø '
79Ø GOSUB 151Ø
8ØØ IF ED THEN LOCATE 23,2 : PRINT ">>==> Can't Load in EDIT mode!"; : RETURN
81Ø IF NOT SV THEN LOCATE 23,2 : PRINT ">>==> Nothing Saved!"; : SV=T% : RETURN
82Ø LOCATE 21,2 : PRINT "F11:Load Font: "; : LINE INPUT FS$
83Ø IF FS$="" THEN RETURN
84Ø ON ERROR GOTO 87Ø : FS$=FS$+".fnt" : BLOAD FS$,WORK.OFF : ON ERROR GOTO Ø
85Ø GOSUB 151Ø
86Ø LOCATE 19,8 : PRINT FS$;"                          "; : ED=F% : GOTO 71Ø
87Ø GOSUB 151Ø : LOCATE 23,2 : PRINT ">>==> F11:Load error";ERL;ERR; : RESUME 82
Ø
88Ø '
89Ø '    Disk Font to Active Font (F7)
9ØØ '
91Ø GOSUB 151Ø
92Ø LOCATE 21,2 : PRINT "F7:Load Font: "; : LINE INPUT FS$ : IF FS$="" THEN RETU
```

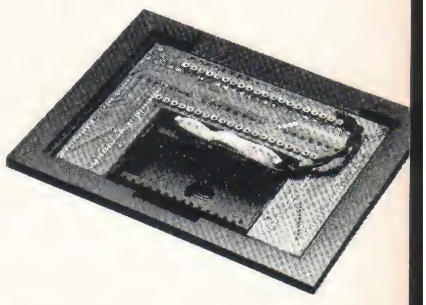
```
RN
93Ø ON ERROR GOTO 95Ø : FS$=FS$+".fnt" : DEF SEG = ACTIVE.SEG : BLOAD FS$,Ø
94Ø DEF SEG : ON ERROR GOTO Ø : GOSUB 151Ø : GOTO 71Ø
95Ø GOSUB 151Ø : LOCATE 23,2 : PRINT ">>==> F7:Load error";ERL;ERR; : RESUME 92Ø
96Ø '
97Ø '    Active Font to Disk Font (F8)
98Ø '
99Ø GOSUB 151Ø
1ØØØ LOCATE 21,2 : PRINT "F8:Save Font: "; : LINE INPUT FS$ : IF FS$="" THEN RET
URN
1Ø1Ø ON ERROR GOTO 1Ø3Ø : FS$=FS$+".fnt" : DEF SEG = ACTIVE.SEG
1Ø2Ø BSAVE FS$,Ø,&H1FFØ : DEF SEG : ON ERROR GOTO Ø : GOSUB 151Ø : GOTO 71Ø
1Ø3Ø GOSUB 151Ø : LOCATE 23,2 : PRINT ">>==> F8:Load error";ERL;ERR; : RESUME 1Ø
ØØ
1Ø4Ø '
1Ø5Ø '    Save a Font routine (F12)
1Ø6Ø '
1Ø7Ø GOSUB 151Ø
1Ø8Ø IF ED THEN LOCATE 23,2 : PRINT ">>==> Can't Save in EDIT mode!"; : RETURN
1Ø9Ø LOCATE 21,2 : PRINT "F12:Save Font: "; : LINE INPUT FS$ : IF FS$="" THEN RE
TURN
11ØØ ON ERROR GOTO 113Ø : FS$=FS$+".fnt" : BSAVE FS$,WORK.OFF,&H1FFØ
111Ø ON ERROR GOTO Ø : GOSUB 151Ø : SV=T%
112Ø LOCATE 19,8 : PRINT FS$;"                              "; : ED=F% : GOTO 71Ø
113Ø GOSUB 151Ø : LOCATE 23,2 : PRINT ">>==> F12:Save error";ERL;ERR; : RESUME 1
Ø8Ø
114Ø '
115Ø '    Display Active Character Font (F1Ø)
116Ø '
117Ø FLOP=FLOP+1 : GOSUB 151Ø : ON FLOP GOTO 118Ø,123Ø,125Ø
118Ø LOCATE 2Ø,1 : PRINT " --------- Ascii Characters ----------  "
119Ø LOCATE 21,1 : PRINT " ABCDEFGHIJKLMNOPQRSTUVWXYZ 123456789Ø  "
12ØØ LOCATE 22,1 : PRINT " abcdefghijklmnopqrstuvwxyz !@#$%^&*()  "
121Ø LOCATE 23,1 : PRINT " <>?:{}_+,./;'[]-=\~|'   Davy Crockett  "
122Ø LOCATE 24,1 : PRINT " --------- Ascii Characters ---------- "; : GOTO 71Ø
123Ø LOCATE 2Ø,2 : FLIP=128 : GOSUB 126Ø : LOCATE 22,2 : FLIP=144 : GOSUB 126Ø
124Ø LOCATE 24,2 : FLIP=16Ø : GOSUB 126Ø : GOTO 71Ø
125Ø LOCATE 2Ø,2 : FLIP=176 : GOSUB 126Ø : FLOP=Ø : GOTO 71Ø
126Ø PRINT USING"### ";FLIP; : FOR K=Ø TO 15 : PRINT CHR$(FLIP+K);" "; : NEXT
127Ø RETURN
128Ø '
129Ø '    Fill in the blocks routine
13ØØ '
131Ø FOR K=Ø TO 15 : LOCATE K+2,3  : B=N(K) : GOSUB 135Ø : NEXT
132Ø FOR K=Ø TO 15 : LOCATE K+2,16 : B=N(K) : GOSUB 135Ø : NEXT
133Ø DEF SEG = ACTIVE.SEG : FOR K=Ø TO 15 : POKE 32+K*2,N(K)
134Ø POKE 64+K*2,N(K) : NEXT : DEF SEG : RETURN
135Ø IF B AND 128 THEN PRINT BS$; ELSE PRINT BR$;
136Ø IF B AND  64 THEN PRINT BS$; ELSE PRINT BR$;
137Ø IF B AND  32 THEN PRINT BS$; ELSE PRINT BR$;
138Ø IF B AND  16 THEN PRINT BS$; ELSE PRINT BR$;
139Ø IF B AND   8 THEN PRINT BS$; ELSE PRINT BR$;
14ØØ IF B AND   4 THEN PRINT BS$; ELSE PRINT BR$;
141Ø IF B AND   2 THEN PRINT BS$; ELSE PRINT BR$;
142Ø IF B AND   1 THEN PRINT BS$; ELSE PRINT BR$;
143Ø RETURN
144Ø '
145Ø '    Dummy RETURN for un-used function keys
146Ø '
```

```
147Ø GOSUB 153Ø : LOCATE 23,2 : PRINT ">>==> Nothing Done"; : RETURN
148Ø '
149Ø '    Miscellaneous Subroutines
15ØØ '
151Ø FOR L=2Ø TO 24 : LOCATE L,1 : PRINT B4Ø$; : NEXT : RETURN
152Ø GOSUB 153Ø : LOCATE 21,1 : PRINT B4Ø$; : RETURN
153Ø LOCATE 23,1 : PRINT B4Ø$; : RETURN :'blank message area
154Ø LOCATE 24,1 : PRINT B4Ø$; : RETURN :'blank auxiliary area
155Ø GOSUB 153Ø : FOR K=2 TO 17 : LOCATE K,3 : PRINT B8$;
156Ø LOCATE K,16 : PRINT B8$; : NEXT
157Ø LOCATE 3,35 : PRINT B5$; : LOCATE 4,35 : PRINT B5$;
158Ø DEF SEG = ACTIVE.SEG : FOR K=Ø TO 15 : POKE 32+K*2,Ø
159Ø POKE 64+K*2,Ø : NEXT : DEF SEG : RETURN
16ØØ '
161Ø '    Procedure Division
162Ø '
163Ø LOCATE  9,13 : PRINT "S"; : LOCATE 1Ø,13 : PRINT "E";
164Ø LOCATE 11,13 : PRINT "L"; : LOCATE 12,13 : PRINT "E";
165Ø LOCATE 13,13 : PRINT "C"; : LOCATE 14,13 : PRINT "T"; : GOSUB 71Ø
166Ø O$=INKEY$ : IF O$ = "" THEN 166Ø
167Ø BYTE%=ASC(O$) : FOR K=Ø TO 15 : N(K)=PEEK(WORK.OFF+BYTE%*32+K*2) : NEXT
168Ø LOCATE  9,13 : PRINT " "; : LOCATE 1Ø,13 : PRINT "E";
169Ø LOCATE  9,13 : PRINT " "; : LOCATE 1Ø,13 : PRINT "E";
17ØØ LOCATE 11,13 : PRINT "D"; : LOCATE 12,13 : PRINT "I";
171Ø LOCATE 13,13 : PRINT "T"; : LOCATE 14,13 : PRINT " ";
172Ø LOCATE 3,34 : PRINT BYTE%; : LOCATE 4,35 : PRINT HEX$(BYTE%);
173Ø GOSUB 131Ø : GOSUB 71Ø : ED=T%
174Ø IF ED THEN GOTO 174Ø
175Ø O$=INKEY$ : IF O$="" THEN GOTO 163Ø ELSE GOTO 175Ø
176Ø '
177Ø '    Initialization Section
178Ø '
179Ø T%=-1 : F%=Ø : ED=F% : X=7 : Y=Ø : CC$=STRING$(3,1) : NC$=STRING$(3,2)
18ØØ ACTIVE.SEG = &HF8ØØ : WORK.OFF = &HEØØØ : GOSUB 158Ø : SV=T%
181Ø DIM N(15) : TL$=CHR$(2Ø1) : TM$=CHR$(2Ø3) : TR$=CHR$(187)
182Ø BL$=CHR$(2ØØ) : BM$=CHR$(2Ø2) : BD$=CHR$(188) : MR$=CHR$(2Ø4)
183Ø ML$=CHR$(185) : B8$="        " : B5$ = "      " : BS$="*" : BR$=" "
184Ø H$=CHR$(2Ø5) : V$=CHR$(186) : H8$=STRING$(8,H$) : H3$=STRING$(3,H$)
185Ø B4Ø$=STRING$(4Ø,32) : FOR K = Ø TO 15 : LOCATE K+2,1
186Ø PRINT HEX$(K);V$;SPC(8);V$;SPC(3);V$;SPC(8);V$; : NEXT
187Ø LOCATE  1, 2 : PRINT TL$;H8$;TM$;H3$;TM$;H8$;TM$;H3$;TR$;
188Ø LOCATE 18, 2 : PRINT BL$;H8$;BM$;H3$;BM$;H8$;BD$;
189Ø LOCATE  5,11 : PRINT MR$;H3$;ML$;B8$;MR$;H3$;BD$;
19ØØ LOCATE 2,28 : PRINT V$; : LOCATE 3,28 : PRINT V$; : LOCATE 4,28 : PRINT V$;
191Ø LOCATE  2,3Ø : PRINT "Character"; : LOCATE 3,3Ø : PRINT "Dec:";
192Ø LOCATE  4,3Ø : PRINT "Hex:"; : LOCATE 19,2 : PRINT "Font:";
193Ø LOCATE  7,26 : PRINT "F1: Set Bit   ";
194Ø LOCATE  8,26 : PRINT "F2: Reset Bit ";
195Ø LOCATE  9,26 : PRINT "F3: End Save  ";
196Ø LOCATE 1Ø,26 : PRINT "F4: End Nosave";
197Ø LOCATE 11,26 : PRINT "F5: Work->Actv";
198Ø LOCATE 12,26 : PRINT "F6: Actv->Work";
199Ø LOCATE 13,26 : PRINT "F7: Disk->Actv";
2ØØØ LOCATE 14,26 : PRINT "F8: Actv->Disk";
2Ø1Ø LOCATE 15,26 : PRINT "F9: System    ";
2Ø2Ø LOCATE 16,26 : PRINT "F1Ø: Display  ";
2Ø3Ø LOCATE 17,26 : PRINT "F11: Load Font";
2Ø4Ø LOCATE 18,26 : PRINT "F12: Save Font";
2Ø5Ø '
```

Give your Model 100

128K RAM

Installs as easily as plugging in a socket



Software included, transfers from bank to bank. Works like main menu! Includes powerful RAM Basic that lets programs store and access data from any other bank.

PCSG says: Satisfaction guaranteed or your money back within 30 days

As amazing as it seems you can upgrade your Model 100 to 128K of RAM in just 60 seconds.

It comes to you right out of the box looking just like the picture. You just open the little compartment on the back of your Model 100 with a quarter and it just pushes right into place. You can then put the cover back in its place.

You then have 4 banks of RAM of 32K each. The additional three banks also work just like your Main Menu.

You push a function key and you are in the second bank. Push again and you are in third, again, then fourth. Press it once again for your original bank.

It has its own built-in NiCad battery that recharges right from the Model 100 and its guaranteed for a full year.

What is really great is that you can copy a file from one bank to another with just a function key.

Each bank is like having another Model 100, and all the built-in programs as well as any snap-in ROM programs appear in all four banks and work the same way. Your widebar cursor moves from file to file and you access any file or run any program just by pressing ENTER.

What lets you copy any file from one bank to another is a snap-in ROM from PCSG called RAM+, that comes at no extra charge. It just pushes right into the little socket in that same compartment with the 96K expansion unit.

Not only does this firmware let you copy a file from bank to bank, but you can make a copy of any file within the same bank instantly with a function key. Great for Lucid spreadsheets!

Copy a file from bank to bank with a function key

You can also rename a file, or kill any file with just a function key. Plus you can do a whole lot of other useful things like setting the date, day and time with function key ease. You even have a function key that lets you use non-Radio Shack printers without having to make those tricky dipswitch settings.

RAM+ lets you cold start any one of your banks without affecting the other three. That means that anytime you want you can clean out a bank's entire memory, but leave intact all the files in the other banks.

What is also fantastic is that you don't have to have the ROM in place to use the additional RAM. Whenever you take out the snap-in ROM it leaves behind a tiny machine code program that lets you switch from bank to bank just by pressing ENTER.

This lets you use your ROM socket to snap-in other ROMS like LUCID spreadsheet, WRITE ROM text processor, or DISK+ ROM file transfer program, and use them in any or all four banks. All of these, by the way, are available from PCSG.

When you are ready to copy a file from one bank to another or use any of the other fantastic functions we talked about you can just snap the RAM+ ROM back into place.

Everybody that has this 128K system in their Model 100 is so excited, because it gives them four times the capacity and all banks work just like the Main Menu.

And what has made a lot of people happy is that the system bus, located in the same compartment, is left free for you to plug in a DVI or the Holmes Engineering PCSG portable disk drive.

The ability to copy a file from bank to bank instantly with a function key, plus all of the other features make this RAM extension truly an engineering masterpiece.

Some people hesitate when they think of installing something, and then others are skeptical that any additional hardware could be as good as the Model 100 itself. That's why we sell these 96K expansions on a 30 day trial. Simply return it within 30 days for a full refund if you are not satisfied. Priced at $425. MC VISA COD.

1-214-351-0564

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd. No. 207, Dallas, Tx. 75229

VISA / MASTER CARD / AM. EXP. / C.O.D.

```
2Ø6Ø '   Initialize Block of characters
2Ø7Ø '
2Ø8Ø LOCATE 2,12 : PRINT CC$; : LOCATE 2,25 : PRINT NC$;
2Ø9Ø LOCATE 3,12 : PRINT CC$; : LOCATE 3,25 : PRINT NC$;
21ØØ LOCATE 4,12 : PRINT CC$; : LOCATE 4,25 : PRINT NC$;
211Ø '
212Ø '   Initialize Function Keys
213Ø '
214Ø ON KEY  (1) GOSUB 36Ø : ON KEY  (2) GOSUB 4ØØ : ON KEY  (3) GOSUB 45Ø
215Ø ON KEY  (4) GOSUB 5ØØ : ON KEY  (5) GOSUB 54Ø : ON KEY  (6) GOSUB 6ØØ
216Ø ON KEY  (7) GOSUB 91Ø : ON KEY  (8) GOSUB 99Ø : ON KEY  (9) GOSUB 66Ø
217Ø ON KEY (1Ø) GOSUB 117Ø : ON KEY (11) GOSUB 79Ø : ON KEY (12) GOSUB 1Ø7Ø
218Ø ON KEY (13) GOSUB 72Ø : ON KEY (14) GOSUB 74Ø : ON KEY (15) GOSUB 75Ø
219Ø ON KEY (16) GOSUB 73Ø : FOR K=1 TO 16 : KEY (K) ON : NEXT : RETURN
```

Listing 2:

```
1ØØ '     ****************************************    Creates LOADFONT.COM
11Ø '     *****  LOADFONT.COM GENERATOR  *****    Version 1.1
12Ø '     ****************************************    13 April 1985
13Ø '
14Ø '     Davy Crockett
15Ø '     58Ø7 Cherrywood Lane
16Ø '     Apartment #1Ø4
17Ø '     Greenbelt, Maryland   2Ø77Ø
18Ø '
19Ø CLS : DEFINT A-Z
2ØØ '
21Ø '    Error Checking Section.
22Ø '
23Ø BYTE$ = "ØØ" : PRINT "LOADFONT Generator Working....."
24Ø WHILE BYTE$ <> "XX"
25Ø   READ BYTE$ : BYTE% = VAL("&H"+BYTE$) : CHECKSUM% = CHECKSUM% + BYTE%
26Ø WEND
27Ø IF CHECKSUM% = 12524 THEN GOTO 32Ø
28Ø PRINT ">>==> Please check the DATA statements and try again!" : STOP
29Ø '
3ØØ '    Generate .COM Section.
31Ø '
32Ø OPEN "LOADFONT.COM" FOR OUTPUT AS #2 : BYTE$ = "ØØ" : RESTORE
33Ø WHILE BYTE$ <> "XX"
34Ø   READ BYTE$ : BYTE% = VAL("&H"+BYTE$) : PRINT #2,CHR$(BYTE%);
35Ø WEND
36Ø CLOSE : PRINT "LOADFONT.COM Created." : STOP
37Ø '
38Ø '   Data Division.
39Ø '
4ØØ DATA 1E,33,CØ,5Ø,BB,ØØ,ØØ,B4,3F,BA,53,Ø1,B9,Ø7,ØØ,CD
41Ø DATA 21,3D,Ø7,ØØ,75,Ø7,AØ,53,Ø1,3C,FD,74,ØF,BB,Ø2,ØØ
42Ø DATA B4,4Ø,BA,5A,Ø1,B9,22,ØØ,9Ø,CD,21,CB,8B,ØE,58,Ø1
43Ø DATA BA,ØØ,ØØ,B8,ØØ,F8,51,1E,8E,D8,B4,3F,CD,21,1F,59
44Ø DATA 3B,C1,74,ØE,BB,Ø2,ØØ,B4,4Ø,BA,7C,Ø1,B9,23,ØØ,9Ø
45Ø DATA CD,21,CB,ØØ,ØØ,ØØ,ØØ,ØØ,ØØ,ØØ,ØD,ØA,3E,3E,3D,3D
46Ø DATA 3E,2Ø,4C,4F,41,44,46,4F,4E,54,3A,2Ø,66,69,6C,65
47Ø DATA 2Ø,6E,6F,74,2Ø,42,53,41,56,45,ØD,ØA,ØD,ØA,3E,3E
48Ø DATA 3D,3D,3E,2Ø,4C,4F,41,44,46,4F,4E,54,3A,2Ø,6C,6F
49Ø DATA 61,64,2Ø,66,69,6C,65,2Ø,65,72,72,6F,72,ØD,ØA,XX
```

Listing 3:

```
        name    LOADFONT
        page    60,120
        title   LOADFONT -- load a new character font
;
;=============================================================*
;                                                             *
;       ****************************        LOADFONT.ASM      *
;       *****  LOADFONT.COM  *****         Version 1.1       *
;       ****************************        13 April 1985     *
;                                                             *
;       Davy Crockett                                         *
;       5807 Cherrywood Lane                                  *
;       Apartment #104                                        *
;       Greenbelt, Maryland   20770                           *
;                                                             *
;       Execute with command:  LOADFONT <fontname.FNT         *
;                                                             *
;=============================================================*
;
;-------------------------------------*
;       M A C R O S                   *
;-------------------------------------*
;
stdio   macro   handle,function,length,buffer
        mov     bx,&handle
        mov     ah,&function
        mov     dx,offset &buffer
        mov     cx,&length
        int     21H
        endm
;
;-------------------------------------*
;       DOS 2.0 Pre-Defined Handles   *
;-------------------------------------*
;
stdin   equ     0000                    ;standard input file
stdout  equ     0001                    ;standard output file
stderr  equ     0002                    ;standard error file
;
cr      equ     13                      ;carriage return
lf      equ     10                      ;line feed
;
;-------------------------------------*
;       Code Segment Section          *
;-------------------------------------*
;
cseg    segment para public 'CODE'
;
        assume  cs:cseg,ds:cseg,es:nothing
;
        org     100H                    ;start .COM at 100H
;
loadf   proc    far                     ;entry point from PC-DOS
        push    ds                      ;push RET segment address
        xor     ax,ax                   ;clear AX and push
        push    ax                      ;  RET offset address
;
;       Check for standard "BSAVE" header
```

```
;
        stdio   stdin,3FH,7,input_buffer
        cmp     ax,7                    ;did we get 7 bytes?
        jnz     file_error_exit         ;no, exit stage right
        mov     al,[input_buffer]
        cmp     al,ØFDH                 ;is this a BSAVE file?
        jz      continue_load           ;yes, go load the bytes
;
;       Bad file, abort the load
;
file_error_exit:
        stdio   stderr,4ØH,file_error_len,file_error_msg
        ret                             ;return to MS-DOS
;
;       File checks out, continue the load
;
continue_load:
        mov     cx,word ptr [input_buffer+5]
        mov     dx,Ø                    ;bit pattern offset
        mov     ax,ØF8ØØH               ;bit pattern segment
        push    cx                      ;save byte count
        push    ds                      ;save current dseg pointer
        mov     ds,ax                   ;install new seg pointer
        mov     ah,3FH                  ;load function code
        int     21H                     ;load the bytes
        pop     ds                      ;restore dseg pointer
        pop     cx                      ;retrieve byte count
        cmp     ax,cx                   ;load all the bytes?
        jz      return_to_dos           ;yes, go to return
;
load_error_exit:
        stdio   stderr,4ØH,load_error_len,load_error_msg
;
return_to_dos:
        ret                             ;return to MS-DOS
;
loadf   endp
;
input_buffer    db      7 dup(?)
;
;---------------------------------------*
;       Error Message Section           *
;---------------------------------------*
;
file_error_msg  db      cr,lf
                db      '>>==> LOADFONT: file not BSAVE'
                db      cr,lf
file_error_len  equ     (this byte)-(offset file_error_msg)
;
load_error_msg  db      cr,lf
                db      '>>==> LOADFONT: load file error'
                db      cr,lf
load_error_len  equ     (this byte)-(offset load_error_msg)
;
;---------------------------------------*
;
cseg    ends
;
        end     loadf
```

PCM

Different Ways to Store Program Data

By Richard A. White

The bread and butter of a computer program is the data it handles. A program exists either to handle data or to cause something to happen based on data it contains or is given. So, it makes sense to discuss ways to store data before anything else.

If you have a Tandy MS-DOS machine and decide to buy the BASIC manual, you get a book that is divided into chapters with a detailed index in the front. There is a chapter on BASIC commands, statements, functions and variables. In the index for that chapter is a long list of each of these topics followed by a page number. Variables are much like entries in an index in that they show the computer how to find specific data. The data itself may be stored somewhere else in the computer's memory.

There are different types of variables since the computer handles data in different ways. The word "hello" is a string of five characters. You might want the computer to store hello and be able to find it later and print it to the screen or printer. To do this, you must assign hello to some letter or string of letters that will represent it in the program and will also tell BASIC that hello is a string of characters. For example, `WORD$="HELLO"` tells the program the variable named `WORD` is to represent `HELLO`. The `$` following `WORD` tells BASIC that `HELLO` is to be handled as a string of characters. Now it seems natural to call variables representing strings of characters "string" variables. In computer jargon, the `$` declares `WORD` to be a string type variable.

String variables vary in length up to 255 characters. They use only the memory required to store the number of characters included in the string, plus the bytes used in the variable table. This is more memory efficient than compiled languages where string variable storage

Richard White has a long background with microcomputers and specializes in BASIC programming. He has authored numerous programs and articles. His work has also appeared in PCM's sister publication, THE RAINBOW.

space must be allocated before compiling the program and string length is selected based on the longest string that might be encountered. However, BASIC is limited to 64K of memory for your program, its data, variable tables and stack, even if you have 640K of memory in your machine. It needs to have economical data storage.

Once a string is assigned to a variable name, BASIC goes into action to put the string into memory and puts the variable name into the variable table along with the string's starting address and its length. The variable table for string data is very much like an index. The data for a particular string variable may be in one place in free memory and change that place if the string is changed, but BASIC keeps its variable table up to date and can always find the string.

Variable names may be any length in GW-BASIC, but only the first 40 characters are saved in the variable table. Still, 40 characters is shear luxury compared to one- or two-character limits in some earlier BASICs. The first character of a name must be a letter, but numbers and the period may be used after the first letter. Now you can use variable names that have true English meaning, making your programs much more readable and understandable.

The various BASIC statement, command and function names are reserved words. These cannot be used as variable names since BASIC has no way of knowing when you want to make PRINT a variable name, you don't want to print something. It gets confused and sends a message explaining why it quit. Reserved words may be part of a longer variable name.

The line 10 VAL=10 is invalid since VAL is a reserved word. The line 10 VALUE=10 is OK since VAL is only part of the name and VALUE is not a reserved word. VALUE is also more meaningful when you read the program.

To prove the rule, there is an exception. FN cannot be used as the first two letters of a variable name except as a call to a user-defined function.

Numbers are handled a little differently by BASIC, but the programmer does not see this. They are stored in the variable table with the variable names that represent them. More important are the different types of numbers that can be stored, why these differences exist and how you can take full advantage of these differences.

Let's start with integers. Integers are whole numbers between -32,768 and 32,767, inclusive. Integers do not have decimal points. Integers provide economical memory usage since they use only two bytes to represent the value and are fast to use. The microprocessor can deal with integers directly, which accounts for the speed.

Integers fit counting like a glove fits a hand. If you want to do something 100 times, counting with whole numbers is all you need. But, you must remember to declare the counting variable as an integer. The % tells BASIC you want an integer variable.

```
10 FOR COUNT%=1 TO 100
20 PRINT COUNT%
30 NEXT
```

In this code, the type of the variable COUNT% is specified when the variable name is used in the program. There is another way.

```
5 DEFINT C
10 FOR COUNT=1 TO 100
20 PRINT COUNT
30 NEXT
```

The DEFINT statement specifies that all variable names that start with the letters listed are integer type. DEFSTR does the same thing with string variable names. A DEFtype statement must precede the first use of a variable name in the program. This means only that BASIC must come across the DEFtype before it sees the variable in use.

```
5 GOTO 10000
10 FOR COUNT=1 TO 100
20 PRINT COUNT
30 NEXT
10000 DEFINT C : GOTO 10
```

In this example, Line 5 sends the program to Line 10000, where BASIC encounters the DEFINT C statement. When BASIC goes to Line 10, it knows that COUNT is an integer even though the DEFINT statement is in a later line. Putting initializing code at the end of the program makes sense only when you understand how BASIC works. Each time BASIC is sent to a different line by a GOSUB or GOTO, it must start at the beginning of the program and look at each line number in order until it finds the one it wants. This takes time and it makes sense to put lines only used once at the end of the program, allowing lines frequently used to be closer to the front.

Finally, we need to use decimal numbers, and BASIC offers two flavors, single-precision and double-precision. Accuracy, speed and memory economy are the issues here. A single-precision number stores seven digits, but only six are accurate. However, it only uses four bytes of memory. The microprocessor cannot deal with a decimal number directly, but must use a procedure built into BASIC that requires many more operations to perform, say, a multiplication than it does in integer math. Since single-precision operations deal with fewer digits than double-precision counterparts, they are faster.

In the line 10 PARTS!=22.765, the ! defines PARTS to be a single-precision number. 10000 DEFSNG P declares PARTS and all other variables that start with 'P' to be single-precision variables whenever they are encountered in the program. If the type of a numeric variable is not specified, single-precision is used (default).

Double-precision numbers are stored with 17 digits, 16 of which are accurate using eight bytes of storage. When using double-precision, you pay a price in speed and memory to gain needed accuracy. If you do not need more than six digit accuracy, don't use double-precision. DEFDBL declares double-precision variables. The # added to the variable name specifies double-precision.

Occasionally, it is necessary to convert a number from one precision to another. Five rules apply:

1) When a numeric value of one precision is assigned to a variable of a different precision, the number is stored as the precision declared for the target variable name. In the following example, TARGET% is an integer, so only the integer part of the assigned value is used (subject to Rule 2).

```
Ok
10 TARGET% = 45.2
20 PRINT TARGET%
RUN
45
Ok
```

2) Rounding rather than truncation occurs when a higher precision number is assigned to a lower precision variable.

```
Ok
10 C = 765.98258437#
20 PRINT C
RUN
765.9826
Ok
```

Single-precision is the default type, so C is single-precision. In this example, a double-precision number is assigned to a single-precision variable. The seventh digit (5) is rounded up. If truncation had occurred, the following '8' would have been ignored and the result would have been 765.9825.

3) If you convert from a lower precision to a higher precision, the resulting higher precision number is the same precision as the source number. If you assign A%, an integer, to B!, a single-precision variable, B! will contain an integer value. However, the higher precision variable may not print like the lower precision one. Observe this example from the IBM PC BASIC manual.

```
Ok
10 A = 2.04
20 B# = A
30 PRINT A;B#
RUN
 2.04  2.039999961853027
```

4) When an expression is evaluated, all the operands (variables and numbers involved) in an arithmetic or relational operation are converted to the precision of the most precise operand. If the expression involves three integers and one single-precision number, the operation is done in single-precision. The result of an arithmetic operation is returned to the degree of precision of the operation. Again, the example is from the IBM PC BASIC manual.

```
Ok
10 D# = 6#/7
20 PRINT D#
RUN
 .8571428571428571
Ok
```

The 6# forces double-precision evaluation, which is returned to a double-precision variable D#. The result could have been returned to a single-precision variable, in which case it is rounded.

```
Ok
10 D = 6#/7
20 PRINT D
RUN
 .8571429
Ok
```

5) Logical operators convert their operands to integers and return an integer result. Given this, it is reasonable to expect the integer range limits, -32768 through 32767, to apply. If you try to use a number outside this range, an "overflow" error occurs.

Each type of variable may be used in an array. An array is a group or table of data referred to by the array name and a number. Each piece of data in an array is called an element and is an individual variable. Array elements can be used in expressions and in any BASIC statement or function like any other variable.

NAMES$(7) is the seventh element of the array NAMES$. If NAMES$(7) is used in a program without a DIM statement, BASIC establishes the NAMES$ string array with a default dimension of 10. This array actually has 11 elements since there is a zero element.

Generally, you need fewer or more elements in an array. Tell BASIC what you need using DIM to dimension the array. 10 DIM NAMES$(50) establishes a one-dimension string array with 51 elements (remember zero). All elements have an initial null value — they are empty strings of zero length.

20 DIM NUMBERS(4,3) establishes a two-dimensioned array of single-precision values called NUMBERS. All elements of the array are initialized to zero.

Specify which element you wish to use by providing the array name and the subscripts of the desired element. 50 NUMBERS(2,3) = 25.7 puts the value in the element, which can be thought of as being in row two of column three of the array. Think of a two-dimensioned array as a table with the first subscript defining the number of rows and the second the number of columns. Remember there is a zero row and a zero column.

Much of the power of arrays comes from the fact that numeric variables can carry the subscript values. In most of my array usage, I take advantage of this fact. Let's try an example.

The variables ROW and COLUMN are counters, so I made them integers out of habit. The elements of the array are single-precision by default. The FOR . . . TO . . . NEXT is a control structure. We will discuss these next month. □

```
                    ARRAY   NUMBERS(4,3)

                                COLUMNS

          NUMBERS(0,0)  NUMBERS(0,1)  NUMBERS(0,2)  NUMBERS(0,3)

          NUMBERS(1,0)  NUMBERS(1,1)  NUMBERS(1,2)  NUMBERS(1,3)

ROWS      NUMBERS(2,0)  NUMBERS(2,1)  NUMBERS(2,2)  NUMBERS(2,3)

          NUMBERS(3,0)  NUMBERS(3,1)  NUMBERS(3,2)  NUMBERS(3,3)

          NUMBERS(4,0)  NUMBERS(4,1)  NUMBERS(4,2)  NUMBERS(4,3)
```

```
Ok
10 DIM NUMBERS(3,2)
20 NUMBERS(0,1) = 1 : NUMBERS (1,0) = 2 : NUMBERS(3,2)=6
30 ROW% = 0 : COLUMN% = 0
40 FOR ROW% = 0 TO 3
50 FOR COLUMN% 0 TO 2
60 PRINT NUMBERS(ROW%,COLUMN%);
70 NEXT COLUMN%
80 PRINT
90 NEXT ROW%
RUN
 0  1  0

 2  0  0
 0  0  0
 0  0  6
Ok
```

PCM

Return Address Label Maker

By Bobby Ballard

I wrote this program a number of years ago simply to print out identical return address labels. It was at first a Color Computer program and now is a GW-BASIC program for my Tandy 1000. It doesn't take long to key into the system; it's very useful and can be easily modified to suit more specific needs. In addition, *Label Maker* contains some pretty useful routines that you may find helpful in writing your own programs.

To use *Label Maker*, just type in the listing and save a copy of it to disk, then press F2 or type RUN. This will show you how the program looks and works in its generic form. To put in a different name, address, city, state and ZIP, or other data like "Return Postage Guaranteed," just edit lines 115-140 and save a copy under a different name. Next time you need a set of these particular labels, load the custom version and run it.

Bobby Ballard is a free-lance writer and the owner of a computer software and consulting firm. He also operates a BBS in Brooklyn. Bobby can be contacted at 1207 Eighth Avenue, Apt. 4R, Brooklyn, NY 11215.

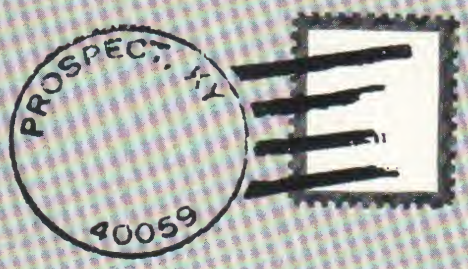
The *Label Maker* should work with just about any DOS computer. There are no graphics commands to worry over. Those with monochrome monitors will want to leave the COLOR commands out, but it may be OK with them left in. Other than that, it's pretty general in the commands used.

To help you understand as much as possible about this program, I've included a list of the variables used and a sample of the printout it produces (Figure 1). Notice the sample printout has a header label that should be aligned to print the first line of a label on your printer. The date and time of the printing is listed on the header label. This information might be useful for logging your computer activity for tax purposes or estimating how many labels are used over a certain period of time.

The Subroutines

Some of the more useful subroutines used in *Label Maker* are the screen and

This simple and versatile program can easily be modified and improved to suit your own needs



label formatting routines, time routine and a simple timer routine.

The subroutine in lines 605-630 is a real time-saver and makes programs look better, too. This routine takes the length of a line of text and computes where to print it on a specified line in order to center it. You don't have to count characters for each line of text you wish to print on the screen just to have the text centered.

This routine is designed to work specifically with an 80 by 25 text screen, but could easily be modified for a 40-character width screen. This is done by entering this subroutine with a value of XX for the video line (1-25) onto which you wish to print and defining SCRN$ as the line of text you wish to center and print.

Centering lines of text on the actual labels is similarly the same process except the medium is more narrow and the TAB function is used instead of the LOCATE command.

Look at the routine in lines 345-350. The constants in this routine are for the 1-inch wide, peel off Tandy labels, product number 26-1328. This routine centers each line on a 34-character width label. If you don't want the label centered, just put a constant into the TAB function and delete the equations for TB in lines 345 and 350.

Another routine is located in lines 700-720. This section of code erases a line of text specified by XX. In this case, the routine handles 78 characters, but this can easily be increased or decreased with the SPACE$ command. This routine passes the value of XX and SCRN$ to the subroutine at Line 610. To call this routine, declare a value of XX for the line to print on and then GOSUB 710. Used in conjunction with the POS function, this can be a valuable subroutine.

While designing this software, I wanted to have the time printed on the screen and constantly updated. The subroutine in lines 850-870 puts the time in the upper left corner of the screen each time a GOSUB 860 is invoked. I put this subroutine call in each of the loops in the program and avoided INPUT and LINE INPUT commands. INPUT statements cause the looping, and the timer,

to stop. The loops are what cause the time to constantly be updated.

One simple subroutine in *Label Maker* appears in one form or another in almost every program I write. It is almost always necessary to delay action in a program at some point. For this task, a routine like the one found in lines 800-820 is set up for use as a timer. The routine is called with a value of XX for the number of ticks you wish to time out on. If XX is equal to 100, this subroutine counts to 100 before returning to the line that called it. Instead of typing in lots of FOR/NEXT loops all over the place, I just define the value of XX and do a GOSUB to use my timer.

The last subroutine is the error-handling routine in lines 900-980. This routine is invoked if the printer is not switched on. So that other program errors do not invoke this routine, it is shut off with the ON ERROR GOTO 0 command in Line 390. This same command is invoked at the beginning of the listing, too, making sure that residual settings do not interfere with this software.

WHILE-WEND

There are several instances of the WHILE and WEND commands in *Label Maker* that are worth looking at: two examples of syntax and one of a nested WHILE/WEND loop.

Line 225 has an example of a WHILE/WEND loop looking for the ENTER key before continuing. While this loop is executing, the program also searches for 0-9 as the only legitimate input to the prompt How many of the above labels are to be printed? Once ENTER is pressed, the program drops down and continues. Note that Line 235 is a nice way to build a string of limited input and update the screen with this string. Also, this loop allows for constant updating of the time on screen using the subroutine at Line 850.

In another example of the WHILE/WEND loop, it is also associated with the INKEY$ command like the aforementioned loop. Look at the code in the lines forming the error-handling routines from lines 900-980. The WHILE/WEND loop here has an additional WHILE/WEND loop nested inside of it. The outer loop first calls INKEY$ command and continues if there is no key pressed. The next WHILE/WEND loop is used to animate the error message in association with the variable FLIP. The outer loop ends with any key press, then executes lines 975-980. The loop in Line 975 clears the bottom of the screen by looping through screen lines 20-25 and repeatedly calling the subroutine at Line 710.

Special Changes

If you want to make custom labels using other features of your printer, just add the code into the listing and save a special copy of that version. For example, Line 335 turns on correspondence mode for the Okidata ML 92/93 printers. Delete this line if you don't own a 92/93 or if it doesn't match your printer. Also, in Line 325, CHR$(24) is the reset code for the 92/93; this ensures that previously used settings don't interfere with the label printing process. Delete or change this, too, if necessary.

Suggested Improvements

Making custom fonts for your particular printer is the most obvious suggestion. After that, you might wish to make the computer indicate how long the printing job took using the TIME$ function previously discussed.

A major add-on project would involve adding DATA statements and ARRAYS to keep all your label data in one program. You could then page through the data using the label graphics as a window and the arrow keys for scrolling through the data. To add a name and address or variation would involve editing the actual data statements.

One way to avoid editing the program is to open a disk file and read the names and addresses from a file that either already exists or one you create yourself. This is much more involved and time consuming than any other way but would be very handy.

Change it, improve on it — the choice is yours. I've given you the tools and a little information to get started. I hope you will experiment and play around with the code.

Peel and Stick

Label Maker is easy to modify and can be useful now and in the future. If you need a quick batch of return labels, this program can do the job. If you need to quickly come up with a different label for one of your clubs or organizations, a business, a friend or family member, *Label Maker* is a useful, quick tool to have in your utility toolbox.

If you have any questions, comments or complaints about *Label Maker* or any other computer topic, contact me in the MS-DOS SIG on Delphi or at my address listed in the bio. I hope you enjoy using, changing and adding to *Label Maker.* And, if you get a chance, peel one off and stick it on a letter to me. □

Figure 1

Label Maker
Variables Table

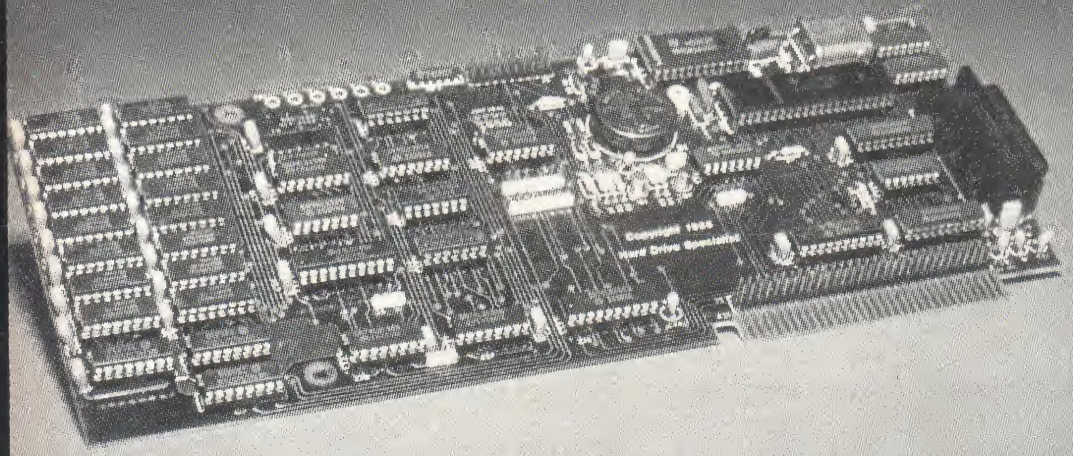
String:	
TITLE$	The border used for the title
EDGE$	Top and bottom edge of label graphics
SIDE$	Side edge of label graphics
SCRN$	Used to center and print text to screen
LC$	Label count
LABELINE$(1-6)	Lines of text for label
Numeric:	
FLIP	Number of times to animate the error message
LC	Label count
LL	Label line count
T	Temporary variable
TIC	Timer beats in subroutine in lines 800-820
XX	Temporary variable for calling subroutines
YY	Used only in subroutine for screen centering (600-640)

The listing:

```
5  '          RETURN ADDRESS LABEL MAKER
```

```
10 '           PRINT ADDRESS LABELS
15 '         FROM T1000 TO ANY PRINTER
20 '         (C) 1986 BY BOBBY BALLARD
25 '         LABEL SIZE   :  1" X 3.5"
30 '         FORM SIZE    :  4" X 12"
35 '
40 '
45 ' Edit lines 105-130 to change the name, address and city, st and zip.
50 '
55 '
60 SCREEN 0,0,0:KEY OFF:CLS
65 ON ERROR GOTO 0
70 TITLE$=STRING$(34,220):EDGE$=CHR$(8)+CHR$(198)+STRING$(34,205)+CHR$(181)+CHR$
(8)
75 SIDE$=CHR$(8)+CHR$(179)+STRING$(34,32)+CHR$(179)+CHR$(8)
80 CLS
85 '       Title to Screen
90 COLOR 4,0 :XX=1:SCRN$=TITLE$:GOSUB 610
95 COLOR 15,0:XX=3:SCRN$="Return Address Label Maker":GOSUB 610
100             XX=4:SCRN$="(c) 1986 by Bobby Ballard" :GOSUB 610
105 COLOR 4,0:XX=6:SCRN$=TITLE$:GOSUB 610
110 '       Define lines of text for label
115 LABELINE$(1) = " "
120 LABELINE$(2) = "NAME"
125 LABELINE$(3) = "ADDRESS"
130 LABELINE$(4) = "CITY, ST  ZIP##"
135 LABELINE$(5) = " "
140 LABELINE$(6) = "Return Postage Guaranteed"
145 '       Put label graphic on Screen
150 COLOR 9,0:XX=9:SCRN$=EDGE$:GOSUB 610
155 FOR XX=10 TO 15:SCRN$=SIDE$:GOSUB 610:NEXT XX:SCRN$=EDGE$:GOSUB 610
160 '       Print lines of label text to Screen
165 COLOR 15,0:XX=10:FOR LL=1 TO 6:SCRN$=LABELINE$(LL):GOSUB 610
170 XX=XX+1:NEXT LL
200 '       Main Loop Starts Here
205 COLOR 14,0:LOCATE 20,(40+INT((LEN(SCRN$)/2))):PRINT SPACE$(LEN(LC$+I$));
210 XX=20:SCRN$="How many of the above labels are to be printed?":GOSUB 610
215 GOSUB 860:LOCATE 1,70:PRINT DATE$
220 YY=0:LC=0:LC$=""
225 I$=INKEY$:WHILE I$ <> CHR$(13)
230    IF I$ < "0" OR I$ > "9" THEN GOSUB 860:GOTO 225 ELSE
235    YY=YY+1:LC$=LC$+I$:LOCATE 20,((40+INT((LEN(SCRN$)/2)))+YY):PRINT I$:GOTO
225
240 WEND
245 LC=VAL(LC$):IF (LC$ = "") THEN 395 ELSE IF (LC < 1) THEN 205 ELSE 250
250 XX=20:GOSUB 710:XX=20:SCRN$="You have chosen to print "+LC$+" labels.":GOSUB
 610
255 XX=21:SCRN$="Correct? [Y,n]":GOSUB 610
260 I$=INKEY$:IF I$="" THEN GOSUB 860:GOTO 260 ELSE
265 IF I$="N" OR I$="n" THEN XX=21:GOSUB 710:GOTO 205 ELSE
270 IF I$="Y" OR I$="y" OR I$=CHR$(13) THEN 280
275 GOTO 255
280 XX=21:SCRN$="Load printer with labels and switch printer ON.":GOSUB 610
285 XX=22:SCRN$="Switch SEL light on and press [ENTER] to start printing.":GOSUB
 610
290 I$=INKEY$:IF I$="" THEN GOSUB 860:GOTO 290 ELSE
295 XX=21:GOSUB 710
300 XX=22:GOSUB 710
305 '          Printer Bells and Whistles
```

The Ultimate Expansion for the Model 1000!
TanPak ™ by Hard Drive Specialist



7 Functions Include:
DMA
Serial Port
Memory up to 512K
Clock/Calendar
Printer Spooler
Memory Disk
Expansion Port for Future Options

Features Include:
Gold Edge Cards
Pre-Tested and Burned In
Full Documentation
Supporting Software
One Year Warranty
Expandable to 512K

TanPak ™ PCM CERTIFICATION SEAL

The TanPak ™ expansion board has been designed to allow expansion beyond the scope of the standard Model 1000. Seven of the most needed functions/features have been combined into one package using only one expansion slot. Your remaining spaces are left free for future expansion needs.

Seven Function/Features on One Board

512K Memory Expansion

Socketed and expandable to 512K. This is done by two banks of memory using either two 64K increments (128K), or 256K increments (256K or 512K). This allows a total of 640K in the Tandy 1000.

Serial Port

Using the same configuration as the Model 1000 port you are assured of complete compatibility as well as being able to configure it as COM1 or COM2.

Clock-Calendar

Quartz-controlled for a high degree of accuracy, featuring a battery backup.

DMA

The DMA (Direct Memory Access) is used on the Model 1000's first memory card. It increases memory speed and is a must for hard drive operation.

Printer Spooler

Use part of your TanPak ™ memory as a printer buffer. Choose the amount of buffer space you need and stop waiting on your printer.

Memory Disk

Use part of your TanPak ™ memory as a RAM drive. With a solid state drive you can store, retrieve, and sort data quickly and easily.

Expansion System

The TanPak ™ was designed with an expansion port that will allow upgrading with additional features when they become available. Some of the possible features are: a second parallel port, a second serial port, mouse, hard drive port, bubble memory, A/D, and D/A as well as many others.

TanPak ™ 0K . . . . . $329.

Call for pricing on memory modules.

TanPak ™ Secondary

If you already have a Model 1000 memory board and do not wish to replace it, the TanPak ™ Secondary is for you. It features Memory up to 256K, Clock/Calendar, Serial Port, Printer Spooler, and Printer Buffer.

TanPak ™ Secondary 0K . . . . . $249.

TanTel ™

Internal 300/1200 baud Modem for the Tandy 1000. Supported by the DeskMate software package.

TanTel ™ . . . . . $249.

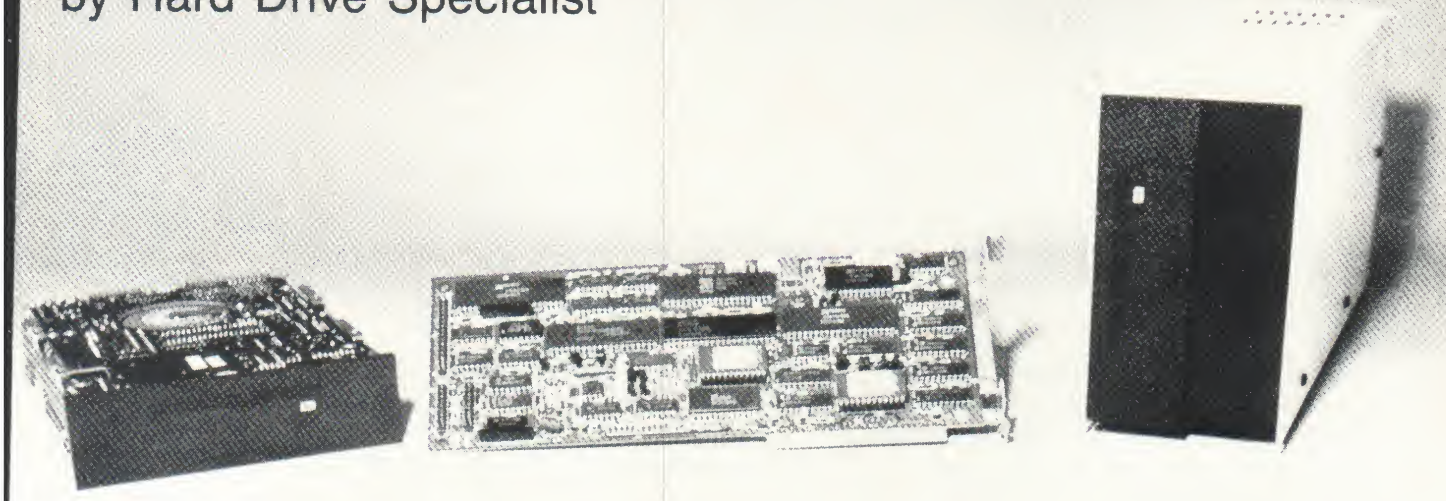
Ordering Information

Use our W.A.T.S. line to place your order via Visa, MasterCard, or Wire Transfer. Or mail your payment directly to us. Any non-certified funds will be held until proper clearance is made. COD orders are accepted as well as purchase orders from government agencies. UPS ground is our standard means of shipping unless otherwise specified. Shipping costs are available upon request. Tandy and Radio Shack are trademarks of the Tandy Corporation. TanPak, and TanTel are trademarks of Hard Drive Specialist, a division of Compukit Corporation.

Hard Drive Specialist
16208 Hickory Knoll
Houston, Texas 77059
1-713-480-6000
Order Line
1-800-231-6671

Megabytes for Mini-Bucks

Tandy 1000 Hard Drives
by Hard Drive Specialist



Shop and compare. Hard Drive Specialist has been building hard drive systems for years and have sold thousands of subsystems to satisfied Radio Shack/Tandy customers. We use the latest state-of-the-art disk drives and controllers. Our drives all use buffered seek logic and plated media to result in almost one-fourth the average access times found on our competitors drives. Plated media results in longer platter life and high resistance to head crashes not found on coated media drives. We utilize high quality construction throughout including heavy duty power supplies and gold edge card connectors. Internal drive systems include an interface card and a half-height hard drive that replaces the top disk drive in both size and power consumption. External units include an interface card, case, power supply and hard drive unit. All units require a memory board with DMA such as the TanPak™ multifunction card, or a Tandy Board, Part #25-1004.

Hard Disk System Features:

- All sizes rated after formatting.
- One secondary drive may be added.
- Error checking and correcting controller.
- Software drivers included.
- Buffered seek drives for improved access time.
- Built in power up diagnostics.
- Plated media for long disk life.
- Gold connectors used throughout.
- 1 year warranty.
- Boots directly from Hard Drive.
- Uses Tandy 1000 MS DOS.

10 Megabyte Internal	**$ 549.**
20 Megabyte Internal	**$ 749.**
42 Megabyte Internal	**$1349.**
10 Megabyte External	**$ 699.**
20 Megabyte External	**$ 899.**
42 Megabyte External	**$1495.**

Ordering Information

Use our WATS line to place your order via Visa, MasterCard, or Wire Transfer. Or, mail your payment directly to us. Any non-certified funds will be held until proper clearance is made. COD orders are accepted as well as purchase orders from government agencies. UPS ground is our standard means of shipping unless otherwise specified. Shipping costs are available upon request. Tandy and Radio Shack are trademarks of the Tandy Corporation. TanPak, is a trademark of Hard Drive Specialist, a division of Compukit Corporation.

Hard Drive Specialist

16208 Hickory Knoll
Houston, Texas 77059
1-713-480-6000

Order Line
1-800-231-6671

```
310 LC$=STR$(LC)
315 ON ERROR GOTO 905
320 XX=21:SCRN$="Printing "+LC$+" labels.":GOSUB 610
325 LPRINT;CHR$(24);"START PRINT"LC"LABELS ":LPRINT DATE$:LPRINT TIME$
330 LPRINT:LPRINT "RETURN ADDRESS LABEL MAKER":LPRINT "(c) 1986 by Bobby Ballard
"
335 LPRINT;CHR$(27);CHR$(49);:        'Okidata 92/93 Corr Mode
340 T=0
345 FOR LL=1 TO 6:TB=INT(LEN(LABELINE$(LL)))
350 LPRINT TAB(INT(17-(TB/2))) LABELINE$(LL)
355 NEXT LL
360 LOCATE 1,1:PRINT TIME$;
365 T=T+1
370 T$=STR$(T)
375 XX=22:SCRN$= "Just printed label number "+T$ : GOSUB 610
380 IF T=LC THEN GOTO 390
385 GOTO 345
390 ON ERROR GOTO 0:FOR ST=1 TO 1000:NEXT ST:XX=23:SCRN$=LC$+" labels have been
printed.": GOSUB 610
395 XX=25:SCRN$="[R]un, [E]nd or back to [D]os?":GOSUB 610
400 LOCATE 1,1:PRINT TIME$;:I$=INKEY$:IF I$="" THEN GOSUB 860:GOTO 400 ELSE
405 IF I$="r" OR I$="R" THEN RUN
410 IF I$="d" OR I$="D" THEN SYSTEM
415 IF I$="e" OR I$="E" THEN COLOR 15,0:SCREEN 0,0,0:CLS:KEY ON:END
420 GOTO 400
500 '       End of Main Loop
600 '       SUBROUTINES BEGIN HERE
605 '       Center Scrn$ and print on line XX
610 YY=INT(40-(LEN(SCRN$)/2))
615 IF YY<=0 THEN YY=1
620 LOCATE XX,YY
625 PRINT SCRN$;
630 RETURN
700 '       Erase command line and return.
710 SCRN$=SPACE$(78):GOSUB 610
720 RETURN
800 '       Timer Routine
810 FOR TIC=1 TO XX:NEXT TIC
820 RETURN
850 '       Print Time$ in Upper Left of Screen
860 LOCATE 1,1:PRINT TIME$;
870 RETURN
900 '       Error handling routine
905 XX=23:SCRN$="PRESS ANY KEY TO CONTINUE":GOSUB 610
910 I$=""
915 WHILE (I$=INKEY$) AND (I$ = "")
920  FLIP=0
925  WHILE FLIP < 2
930     XX=21:SCRN$="*/*/*/*/*/* PRINTER ERROR *\*\*\*\*\*":GOSUB 610
935     XX=21:SCRN$="*\*\*\*\*\* PRINTER ERROR */*/*/*/*/*":GOSUB 610
940     GOSUB 860
945     FLIP=FLIP+1
950  WEND
955  XX=21:GOSUB 710
960  XX=50:GOSUB 810
965  GOSUB 860
970 WEND
975 FOR XX=20 TO 25:GOSUB 710:NEXT XX
980 RESUME 205
```

PCM

Turbo Powered *DeskMate*

By Bobby Ballard

In the past I've discussed the differences between the way *DeskMate* saves text files and DOS reads them. This month I have a program for you to use in converting *DeskMate* files to DOS text files. It's written in BASIC and has a few added bells and whistles thrown in for good measure. In addition, I want to convince you to "turbo power" *DeskMate* on your system. It's free and easy if you have the memory.

If you've followed my column the past few months, you're aware of the fact that *DeskMate* saves text files without line feeds at the end of each line of text, only carriage returns. DOS needs both to correctly read a file to the screen using the TYPE command. In addition, there are a number of programs, in public domain, that must have standard DOS files in order to work properly. If the *DeskMate* files can be converted, you can put many of these programs to work on your files.

The difference between the two files can cause some of the DOS text filters to behave poorly, if at all; and some of these filters are valuable public domain programs that would be nice to put to use. I thought if I could convert the files to standard DOS text files, I could use some of this wonderful public domain software. Plus, a converted file can still be read into *DeskMate* without any problems; *DeskMate* just ignores the line feeds.

The best approach would be to write the conversion program in a language that is readily accessible, BASIC. So I went to work and found that the solution was quite simple to implement. Much simpler than I, at first, imagined.

The original thinking was that BASIC would be easy to use and I would really only have to open two files, read in the old file, append the necessary ASCII codes and write the line back out to the second file. It then occurred to me that BASIC writes its files with the CR/LF combination common to DOS text files. That eliminated the need to do any appending, making this much easier than first guessed.

Once you've processed a text file through the *DeskMate* Text Filter, you can use the TYPE command to get a normal screen display. If you haven't tried this or seen this problem before, try the following. Put a *DeskMate* disk with text files (ending in .DOC) in a drive and, from the DOS command level, enter the following line:

```
TYPE filename.DOC
```

Notice that all of the lines print on just a few lines. This demonstrates the missing line feeds. Once you process one of these files through the Text Filter, it will print to the screen properly and behave more like a standard DOS text file.

Type in the listing exactly as it is printed. Please don't try to make any modifications before you've gotten this version up and running correctly. The listing is not that long and has been fully commented. Let me point out that it has very little, if any, error trapping. Make sure of the filenames you select and make sure there is room on the disk for the new file, which will be a little larger than the *DeskMate* file. Also make sure the printer is switched on before you elect to do a printout.

Once you've gotten the *DeskMate* Text Filter typed in and debugged, run it and follow the prompts. I've included some instructions in the program but it's so simple to follow and use I doubt you will need them. I included the instructions so you don't have to refer back to this issue to remind you how or what it does.

The Text Filter only needs two filenames to get the job done. The first filename should be the name as saved under *DeskMate*. The second filename should be the name you wish the new converted file to be called. Remember, *DeskMate* will not recognize it unless it has the extension .DOC as part of the name. If you will not be reading the file

Bobby Ballard is a free-lance writer and the owner of a computer software and consulting firm. He also operates a BBS in Brooklyn. Bobby can be contacted at 1207 Eighth Avenue, Apt. 4R, Brooklyn, NY 11215.

back into *DeskMate*, you could use the same filename with the extension .TXT or .DOS, for example.

In addition to converting *DeskMate* files, the Text Filter has some added features that you might wish to put to use. While its main output is to diskette, you may also obtain hardcopy from your printer or you can get the Text Filter to scroll the text to your video screen. Both of these features can be toggled on and off before doing a conversion.

If you use any of the other output features — printer or screen — the Text Filter operates slower. For fastest operation, leave the alternate output features off. Of course, the faster your printer operates, the faster the throughput of the Text Filter when using printer output. In any case, the Text Filter reports the line number it is converting on the bottom left corner of your video screen.

When the process is finished, the Text Filter reports the total number of lines converted and displays the two filenames used in the conversion. You are then presented with the choice to run the program again, end the program and go back to BASIC or to exit the program and BASIC and return to DOS.

RAM Power Your *DeskMate*

Many of you are running systems that have been upgraded to contain 384K, 512K or 640K RAM. If you are running a system with lots of RAM, you can opt to use part of that RAM as another disk drive. To some of you this is not news.

There are several public domain programs that create a RAM disk in memory. I downloaded one from a local BBS here in New York. To initialize this program, you include the following line in your CONFIG.SYS file:

```
DEVICE=RAMDISK.SYS nnn
```

with *nnn* set to the size of the RAM disk you wish to use. Some of the other programs allow you to specify sector size and other parameters. There is one on Delphi also in the MS-DOS SIG.

If you've used a RAM disk before, you know what I'm saying when I point out the major feature of a RAM disk is decreased disk access time resulting in much faster operation of many programs. *DeskMate* is no exception. When you set up your AUTOEXEC.BAT file to transfer *DeskMate*'s major files to your RAM disk, you get *DeskMate* running faster than it would on a hard disk!

With *DeskMate* running at high speed, you jump from section to section in a split second. The response in time is literally milliseconds. Accessing files on floppies or a hard disk will not benefit from using the RAM disk, still, the overall operation from section to section speeds up significantly. I use a RAM disk exclusively for some software and *DeskMate* is one. I no longer wait on the overlays to load when moving about and I do my saves to RAM as well, then move those files with today's date to the physical diskette when the session is over.

You should take advantage of the RAM disk software available and really "turbo power" your copy of *DeskMate*. If you do not have a modem and cannot download a copy of a RAMDISK.SYS file, send me a diskette and a return mailer with postage, and I will send you a copy of the ones I have.

Get On-Line with XMODEM

I know many of you would like to get a copy of a telecommunications program that supports XMODEM and can't really download a copy with the Telecom section of *DeskMate*. Send me a diskette with a return mailer, self addressed, and stamped with the correct postage and I will send you a copy of *Procomm*.

Procomm, from PIL Software, is the best telecommunications program I've ever seen and it is in *public domain*. You are requested to send authors a donation and you will want to do so after you've seen *Procomm* in operation (which is within seconds — it's so easy to use). Documentation is included in a text file that you print on your printer. I understand the limitations of *DeskMate* and you will soon, if you haven't already, wish to move to a better and more powerful program.

In addition to XMODEM, *Procomm* supports YMODEM, YMODEM Batch and Kermit transfer protocols. It is fast, easy to use and well documented. It makes nice use of "exploding" windows, has every feature for which you could wish including various terminal emulation modes for mainframe communications, macros, Kermit file server support and much more. It's the best.

EOF

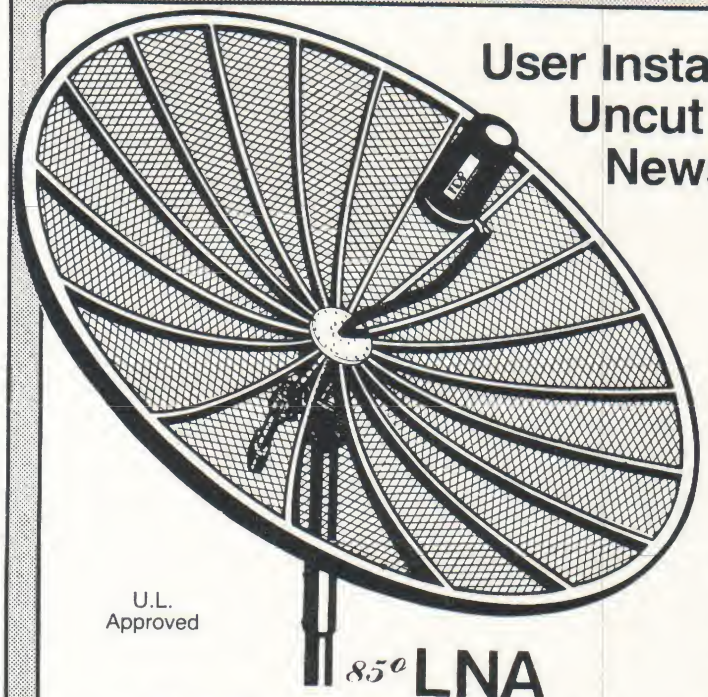
We've covered a lot of areas this month. If you have any questions or comments, send them along. Remember, if you are in need of a communications program or RAMDISK.SYS, send along the diskette with the appropriate enclosed materials mentioned above and I will help you out. Please donate to those whose public domain software you use. Include a self-addressed stamped business-size envelope when writing me if expecting a reply. Thanks, and I'll see you next month. □

Variables Table for Text Filter

Variables	Line Numbers
FALSE	120, 140, 160, 2080, 2120
I$	230, 240, 250, 260, 300, 350, 360, 370, 380, 390, 570, 580, 590, 600, 610, 1060, 1070, 1080, 1090, 1100, 2590, 2600
INFILE$	460, 750, 930
LINENUMBER	110, 790, 810, 1000
OUTFILE$	520, 760, 960
PO$	170, 320, 2110, 2120
PRINTER	160, 390, 830, 2110, 2120
SCRN$	200, 280, 320, 420, 440, 480, 500, 540, 930, 960, 980, 1000, 1030, 2020, 2040, 2160, 2190, 2220, 2290, 2320, 2350, 2380, 2410, 2470, 2500 2560
SO$	150, 320, 2070, 2080
STANDARD	140, 380, 820, 2070, 2080
T	780, 820, 830, 840
TRUE	130, 820, 830, 2070, 2110
XX	190, 270, 310, 410, 440, 470, 500, 530, 920, 950, 970, 990, 1020, 2030, 2070, 2080, 2110, 2120, 2280, 2310, 2340, 2370, 2400, 2430, 2440, 2460, 2490, 2520, 2530, 2550
YY	440, 500, 970, 2020, 2030, 2440, 2530

Discover a Whole New World of TV Entertainment!

SATELLITE SYSTEM



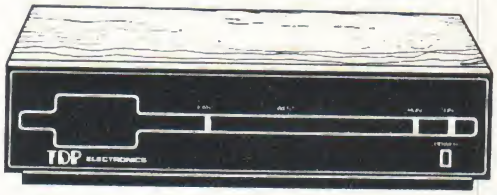
U.L. Approved

85° LNA

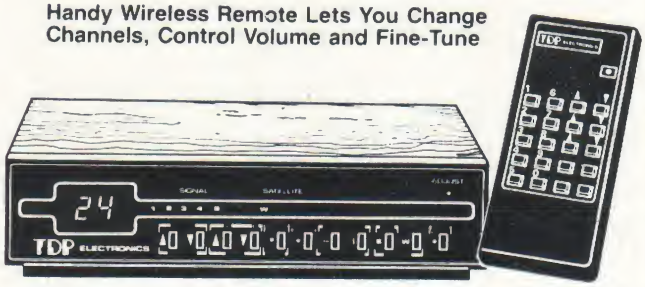
User Installable! Over 100 Channels! Uncut Movies! Specials! Sports! News! Weather! Music Videos!

Entire System **$1519**

- Totally Automatic System With 8½ -Ft. Dish, Receiver and Wireless Remote Control
- Comes With All Accessories Needed for Complete Do-it-Yourself Installation
- Descrambler Ready



Handy Wireless Remote Lets You Change Channels, Control Volume and Fine-Tune



Now you can enjoy it all! 24-hour non-stop entertainment that you simply can't get from local TV, cable, and even video tape! You'll see the hottest stars, recent uncut and uninterrupted movies, the ultimate in sports coverage—and the best part is, there are no monthly fees to pay. You'll see exciting family-oriented programs, plus music videos, FM broadcasts and movies with provision for stereo, and 24-hour news coverage on both TV and radio. It all adds up to out-of-this-world TV viewing at a down-to-earth price. Come in today and discover how easy it is to enjoy satellite programming! Includes complete installation manual.

(817) 825-4027

NOCONA ELECTRONICS • Box 593 • Nocona, TX 76255

UPS Shipping

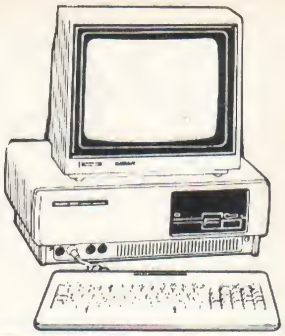
The listing:

```
10 '          DeskMate Text Filter Ver. 1.0
20 '          (c) 1986 by Bobby Ballard
30 '          February 1986 - NYC
40 '
50 CLS
60 SCREEN 0,0,0
70 WIDTH 80
80 KEY OFF
90 BEEP ON
100 SOUND ON
110 LINENUMBER=0
120 FALSE=0
130 TRUE=-1
140 STANDARD=FALSE
150 SO$="on/OFF"
160 PRINTER=FALSE
170 PO$="on/OFF"
180 GOSUB 2150
190 XX=10
200 SCRN$="Do you need instructions? y/N"
210 GOSUB 2020
220 BEEP
230 I$=INKEY$
240 IF I$="" THEN 230
250 IF I$=CHR$(13) OR I$="N" OR I$="n" THEN 270
260 IF I$="Y" OR I$="y" THEN GOSUB 2260:GOTO 270
270 XX=10
280 SCRN$="Select Output below and press [ENTER]"
290 GOSUB 2020
300 I$=""
310 XX=25
320 SCRN$="[S]creen Output "+SO$+"   [P]rinter Output "+PO$
330 GOSUB 2020
340 BEEP
350 WHILE+ (I$<>CHR$(13))
360 I$=INKEY$
370 IF I$<>"S" AND I$<>"s" AND I$<>"P" AND I$<>"p" THEN 350 ELSE
380 IF I$="S" OR I$="s" THEN STANDARD=NOT(STANDARD):GOSUB 2070 ELSE GOSUB 2070
390 IF I$="P" OR I$="p" THEN PRINTER=NOT(PRINTER):GOSUB 2110 ELSE GOSUB 2110
400 WEND
410 XX=10
420 SCRN$="Name of input (DeskMate) file:  "+STRING$(12,32)
430 GOSUB 2020
440 LOCATE XX,(YY-12+(LEN(SCRN$)))
450 BEEP
460 LINE INPUT;INFILE$
470 XX=XX+1
480 SCRN$="Name of output (MS-DOS) file:  "+STRING$(12,32)
490 GOSUB 2020
500 LOCATE XX,(YY-12+(LEN(SCRN$)))
510 BEEP
520 LINE INPUT;OUTFILE$
530 XX=XX+2
540 SCRN$="Is the above information correct? Y,n "
550 GOSUB 2020
560 BEEP
```

```
57Ø I$=INKEY$
58Ø IF I$="" THEN 57Ø
59Ø IF I$=CHR$(13) THEN 72Ø ELSE
6ØØ IF I$="Y" OR I$="y" THEN 72Ø ELSE
61Ø IF I$="N" OR I$="n" THEN RUN
7ØØ '
71Ø '            File Transfer Loop
72Ø '
73Ø LOCATE 25,1
74Ø PRINT"Line # ";
75Ø OPEN "I",#1,INFILE$
76Ø OPEN "O",#2,OUTFILE$
77Ø IF EOF (1) THEN 86Ø
78Ø LINE INPUT #1,T$
79Ø LINENUMBER=LINENUMBER+1
8ØØ LOCATE 25,7
81Ø PRINT LINENUMBER;
82Ø IF STANDARD=TRUE THEN LOCATE 24,1:PRINT T$ ELSE
83Ø IF PRINTER=TRUE THEN LPRINT T$ ELSE
84Ø PRINT #2,T$
85Ø GOTO 77Ø
86Ø CLOSE #1
87Ø CLOSE #2
88Ø '            Report End of Process and Results
89Ø '            Prompt for More?, End?, DOS?
9ØØ CLS
91Ø GOSUB 215Ø
92Ø XX=1Ø
93Ø SCRN$="Input (DeskMate) file: "+STRING$(1Ø,32)+INFILE$
94Ø GOSUB 2Ø2Ø
95Ø XX=XX+2
96Ø SCRN$="Output (MS-DOS) file : "+STRING$(1Ø,32)+OUTFILE$
97Ø LOCATE XX,YY
98Ø PRINT SCRN$
99Ø XX=XX+2
1ØØØ SCRN$="Total Number of Lines Transferred"+STRING$(6,32)+STR$(LINENUMBER)
1Ø1Ø GOSUB 2Ø2Ø
1Ø2Ø XX=25
1Ø3Ø SCRN$="[M]ore, [D]os, [E]nd"
1Ø4Ø GOSUB 2Ø2Ø
1Ø5Ø BEEP
1Ø6Ø I$=INKEY$
1Ø7Ø IF I$="" THEN 1Ø6Ø
1Ø8Ø IF I$="M" OR I$="m" THEN RUN
1Ø9Ø IF I$="D" OR I$="d" THEN BEEP:SYSTEM
11ØØ IF I$="E" OR I$="e" THEN BEEP:CLS:SCREEN Ø,Ø,Ø:KEY ON:END
1111Ø '            END OF MAIN PROGRAM LOOP
2ØØØ '            SUBROUTINES BEGIN HERE
2Ø1Ø '            Print SCRN$ on line XX on center.
2Ø2Ø YY=(4Ø-INT(LEN(SCRN$)/2))
2Ø3Ø LOCATE XX,YY
2Ø4Ø PRINT SCRN$;
2Ø5Ø RETURN
2Ø6Ø '            Set and Reset SO$
2Ø7Ø IF STANDARD=TRUE THEN SO$="ON/off":LOCATE XX,32:PRINT SO$;:RETURN ELSE
2Ø8Ø IF STANDARD=FALSE THEN SO$="on/OFF":LOCATE XX,32:PRINT SO$;:RETURN
2Ø9Ø RETURN
21ØØ '            Set and Reset PO$
211Ø IF PRINTER=TRUE THEN PO$="ON/off":LOCATE XX,58:PRINT PO$;:RETURN ELSE
```

```
212Ø IF PRINTER=FALSE THEN PO$="on/OFF":LOCATE XX,58:PRINT PO$;:RETURN
213Ø RETURN
214Ø '              Title Screen
215Ø XX=1
216Ø SCRN$="< < < < DeskMate Text Filter > > > >"
217Ø GOSUB 2Ø2Ø
218Ø XX=XX+1
219Ø SCRN$="MS-DOS Version 1.Ø"
22ØØ GOSUB 2Ø2Ø
221Ø XX=XX+1
222Ø SCRN$="(c) 1986 by Bobby Ballard"
223Ø GOSUB 2Ø2Ø
224Ø RETURN
225Ø '            On Line Documentation
226Ø CLS
227Ø GOSUB 215Ø
228Ø XX=5
229Ø SCRN$="INSTRUCTIONS"
23ØØ GOSUB 2Ø2Ø
231Ø XX=8
232Ø SCRN$="This program converts DeskMate text files to MS-DOS text files.  You
 supply the"
233Ø GOSUB 2Ø2Ø
234Ø XX=XX+1
235Ø SCRN$="input (DeskMate) filename and an output (MS-DOS) filename at the pro
mpts and it"
236Ø GOSUB 2Ø2Ø
237Ø XX=XX+1
238Ø SCRN$="does the rest.  Before each file the program prompts you for Screen
and Printer"
239Ø GOSUB 2Ø2Ø
24ØØ XX=XX+1
241Ø SCRN$="output settings.  The program runs faster with  all output OFF.  How
ever, using"
242Ø GOSUB 2Ø2Ø
243Ø XX=XX+1
244Ø LOCATE XX,YY
245Ø PRINT"these swithces allows text to be scanned or printed as it is processe
d."
246Ø XX=XX+2
247Ø SCRN$="NOTE:  There is very little error trapping in this version of DeskMa
te File     "
248Ø GOSUB 2Ø2Ø
249Ø XX=XX+1
25ØØ SCRN$="        Filter.  Make sure of the file names you select and that your
 printer is"
251Ø GOSUB 2Ø2Ø
252Ø XX=XX+1
253Ø LOCATE XX,YY
254Ø PRINT"        ON and ready before selecting printer output ON."
255Ø XX=25
256Ø SCRN$="< Press any key to continue >"
257Ø GOSUB 2Ø2Ø
258Ø BEEP
259Ø I$=INKEY$
26ØØ IF I$="" THEN 259Ø ELSE
261Ø CLS
262Ø GOSUB 215Ø
263Ø RETURN                                                              PCM
```

PERRY COMPUTERS



CALL
1-800-248-3823

COLOR COMPUTERS	LIST PRICE	OUR PRICE
26-3127 64K Extended Color Computer 2	$199.95	$169.00
26-3134 16K Standard Color Computer 2	119.95	99.00
26-3131 Thinline Disk Drive 0 for Color Computer	299.95	240.00
26-3018 Extended Basic Kit	39.95	36.00
26-3030 OS-9 With Editor Assembler	69.95	59.50
26-3012 Deluxe Joystick	29.95	25.00
26-1208 CCR-81 Recorder	59.95	50.00
30001170 VIP Integrated Library	149.95	139.00
30001110 VIP Writer	69.95	59.00
30001210 Telewriter 64 Tape	49.95	42.00
30001220 Telewriter 64 Disk	59.95	49.00
30009110 Botek Interface		59.00
TANDY 3000		
25-4001 Tandy 3000 One Disk 512K	$2599.00	$1900.00
25-4010 Tandy 3000 One Disk & 20 Meg HD 512K	3599.00	2600.00
25-3046 Deluxe Text Display Adapter	249.95	205.00
26-3047 Deluxe Graphics Display Adapter	499.95	395.00
25-4030 512K to 2 Meg Expansion Board	499.00	395.00
25-4033 Math Co-Processor for 3000	399.95	339.00
25-4050 1.2 Meg Disk Drive Kit for 3000	299.95	255.00
25-4060 Hard Disk Controller Board for 3000	499.00	425.00
25-4062 20 Meg Hard Disk Kit	799.00	679.00
25-4101 MS-DOS 3.1/Basic/DeskMate	99.95	85.00
26-5111 VM-1 Monochrome Monitor	199.95	165.00
26-5112 CM-1 Color Monitor	599.95	510.00
TANDY 100, 200, 600		
26-3901 Tandy 600 Port Comp 32K 3½ DD	$1599.00	$1195.00
26-3904 Basic Rom for Tandy 600	129.95	110.00
26-3910 92K RAM Upgrade for Tandy 600	399.95	340.00
26-3860 Tandy 200 Port Comp 24K	799.00	650.00
26-3866 24K RAM Memory Exp Chip for 200	199.95	170.00
26-3802 Tandy 100 Port Comp 24K	499.00	425.00
26-3816 8K RAM Upgrade for Tandy 100	49.95	42.00
26-3808 Port 3½ Disk Drive for T100 T200	199.95	169.00
26-3815 Portable Disk Case	24.95	21.00
26-3805 Acoustic Coupler	39.95	34.00
26-1409 Printer Cable	14.95	12.70
26-1410 Modem Cable	19.95	17.00
MODEL 4D		
26-1070 Model 4D 64K 2 FD w/DeskMate	$1199.00	$895.00
26-1122 64K Memory Expansion	69.95	65.00
26-1123 Model 3 to Model 4 Upgrade Kit	399.00	385.00
26-1134 Model 4 HD Kit for 15 Meg	79.95	67.95
26-1530 Multiplan Model 4	199.00	169.00
26-1595 Super Scripsit Model 4	199.95	169.00
26-1608 DeskMate for Model 4	195.00	175.00
26-1635 Profile 4 Model 4	249.95	200.00
26-2216 CP/M Plus Model 4	149.00	127.00
26-2231 Double Duty Utility	69.95	58.00
MODEL 6000 COMPUTERS		
26-6021 Model 6000 512K 2 FD Comp	$4499.00	$3125.00
26-6022 Model 6000 512K 1 FD 15 Meg HD	5499.00	3795.00
26-6019 Model 6000 256K Memory Kit	199.95	185.00
26-6052 DT-100 Data Terminal	795.00	600.00
26-1245 10 Meg Disk Cartridge System	2195.00	1750.00
26-6018 6000 Interface Kit for 26-1245	119.95	102.00
25-3022 1000/1200/3000 Interface Kit for 26-1245	199.95	160.00
26-4155 15 Meg Hard Drive Primary	1595.00	1355.00
26-4171 35 Meg Hard Drive Primary	2995.00	2545.00
26-4173 70 Meg Hard Drive Primary	4295.00	3350.00
26-4157 Installation Kit for Primary HD	349.00	295.00

All prices and offers may be changed or withdrawn without notice. Advertised prices are cash prices. C.O.D. accepted ($10.00 charge per carton on C.O.D. Call for further C.O.D. information.) M.C., Visa, add 2%. AX, add 3%. All non-defective items returned will be subject to 10% restocking fee. Defective items require return merchandise authorization. Call for R.M.A. Number before returning. Delivery is subject to product availability.

RADIO SHACK PRINTERS	LIST PRICE	OUR PRICE
26-1276 DMP-105 Dot Matrix Printer	$ 199.00	$ 169.00
26-1275 TRP-100 Portable Thermal Printer	299.00	230.00
26-1280 DMP-130 100 cps Triple Mode Printer	349.95	285.00
26-1268 CGP-220 Color Ink-Jet Printer	699.00	595.00
26-2830 PC-695 8-Color Digital Plotter	799.00	680.00
26-1277 DMP-430 24 Wire Matrix Printer	899.00	635.00
26-1270 DWP-510 43 cps Daisy Wheel	1495.00	1225.00
26-1274 DMP-2100P 24 Dot Matrix Printer	1495.00	1195.00
26-1279 DMP-2200 HiSpeed Matrix Printer	1695.00	1440.00
26-1269 PT-64 Printer Controller	249.95	210.00
26-1498 SW-302 Printer Switch	119.95	100.00
26-1447 Bi-Directional Tractor for DWP 510	219.95	186.95
TANDY 1200, 2000		
25-3000 Tandy 1200 One FD & 10 Meg HD	$1999.95	$1525.00
25-3001 Tandy 1200 Two FD 256K	1499.00	1200.00
25-3010 VM-3 Green Monitor	219.95	185.00
25-1022 CM-10 Color Monitor	459.95	380.00
25-3043 Graphics Display Adapter	299.00	185.00
25-3044 Graphics Master	695.95	540.00
25-3061 Captain Multifunction Board	599.95	475.00
25-3020 TCS-100 Tape Cartridge System	1999.00	1555.00
25-3021 TCS-100 Interface Kit 1000/1200	149.95	120.00
25-3130 MSDOS/BASIC	89.95	76.50
26-5103 Tandy 2000 Two Disk	1599.00	1225.00
26-5104 Tandy 2000 One Disk & 10 Meg HD	2499.00	1885.00
TANDY 1000		
25-1000 Model 1000 128K 1 FD & DeskMate	$ 999.95	$ 705.00
25-1001 Model 1000 256K 1 FD 10 Meg HD	1999.00	1475.00
25-1003 300-Baud Modem Board 1000/1200	149.95	125.00
25-1005 Disk Drive Expansion 1000	199.95	170.00
25-1006 RS-232C Interface 1000/1200	99.95	85.00
25-1013 1200-Baud Modem Board 1000/1200	299.95	230.00
25-1025 10 Meg Hard Disk Drive	699.95	559.00
25-1007 Hard Disk Control Board	299.95	255.00
25-1021 CM-4 Color Monitor	299.95	225.00
25-1022 CM-10 Color Monitor	459.95	380.00
26-3211 VM-2 Monochrome Monitor	149.95	125.00
30051030 PJB Multifunction Board 512K		310.00
EPSON PRINTERS		
20001015 Homewriter 10	$288.00	$215.00
20001025 LX-80 Dot Matrix Printer	299.00	225.00
20001035 FX-85 Dot Matrix Printer	499.00	345.00
20001050 HI-80 4 Pin Plotter	599.00	390.00
20002010 RX-100 Dot Matrix Printer	499.00	400.00
20002030 FX-286 Dot Matrix Printer	749.00	510.00
20003020 LQ-800 24 Pin Head Dot Matrix	799.00	565.00
20003030 LQ-1000 24 Pin Head Dot Matrix	995.00	695.00
10081010 DX-10 Daisy Wheel Printer	299.00	255.00
10081020 DX-20 Daisy Wheel Printer	459.00	390.00
10081030 DX-35 Daisy Wheel Printer	899.00	765.00

For Technical Questions and Information on our complete line of computer accessories and current prices:

CALL 1-517-625-4161
FOR ORDERS ONLY CALL 1-800-248-3823

Mon., Wed. & Fri. 9-9, Tues. & Thurs. 9-5, Sat. 9-3

124 S. MAIN ST, PERRY, MICH. 48872

PCMfest Reporter

Vol. 1, No. 2
May 1986

Prospect, Kentucky

Editor: Danny Humphress
Writer: Bruce Warner

PCMfest in Palo Alto

For the first time on the west coast, PCM hosted PCMfest, February 14-16, 1986. Despite record rainfalls in the area, computer enthusiasts came out to see what's new for their favorite Tandy computer.

The combined RAINBOWfest/PCMfest covers virtually every computer Tandy makes, from the low-end Color Computer to the high-end Tandy MS-DOS machines. The most exciting thing was watching the show exhibitors pushing the Tandy line of computers beyond the expectations of Tandy or even their three-letter competitor.

Portable Computer Support Group

Lack of RAM, expense of portable disk drives, no powerful spreadsheets. These all sound like the typical lament of Model 100 owners. Now there's an answer.

The Portable Computer Support Group (PCSG) was showing off their answers to all these complaints and more. The Tandy line of portable computers is being expanded to make sure they answer your needs. Their inclusion of the *6-ROM Back* is making the Tandy Model 100 a truly portable computer. *6-ROM Back* can be populated with 128K RAM, *WriteROM* word processor, or Lucid electronic spreadsheet and comes with its own 30-hour power supply. No need for external power supply (except for periodic recharging), full-powered applications software for the most commonly used programs and six plug-in ROMs online at the turn of a switch.

Star Software that shines in the night

For years we've thought of accountants as people who are shut off in a room somewhere adding and subtracting columns of figures. Star Software may be out to change that.

Showing off their full line of accounting software, a variety of programs that appears to be a reliable partner for anyone who has to keep track of massive figures. Maybe with the help of Star Software, we'll change our impression of accountants day.

Delphi online

One of the real advantages of computers is their ability to transfer information over telephone lines. In minutes, you've got an exact copy of your document on the other end of the country.

If you have a sophisticated enough system, you can have several people get together for a conference, and local connect time is cheaper than long distance calls when you have five or more people connected.

Delphi offers you all this and more. During Palo Alto, the folks from Delphi were showing their stuff, with both a Tandy 1000 and a Color Computer online.

Micro Mainframe makes 1000 bigger and better

There is always something eye-catching at a computer show. It can be the title of a program, the look of a monitor or the name of a company, and in Palo Alto the name won out. The name is Micro Mainframe. It seems contradictory, but when you change your Tandy 1000 microcomputer into the mainframe of just four or five years ago, Micro Mainframe doesn't sound so strange.

The Micro Mainframe 4N1 Board includes 512K memory (taking your 1000 to 640K), RS-232C serial port, DMA controller and clock-calendar/mouse. Add to the 1000 a 10-meg hard disk or a video digitizer and you just might be pushing the limit of the entire 8086/88 family of microprocessors.

Popcorn — software that really jumps

One of the most enjoyable things around our home is going out to rent a videotape, coming home and eating popcorn while watching the movie.

For Popcorn Software, they're popping, too, and no wonder. For the first time, I was able to watch MS-DOS software at a reasonable price. Not $300 to $800, but every package listing for under $100, and most for under $50. Perhaps the Color Computer software prices had finally made it to the MS-DOS community.

The three packages shown at the booth were *Popcorn Writer*, *Popcorn Desktop* and *Popcorn Misspeller's Dictionary*. Each is at least similar to another much more expensive software package on the market.

Hard Drive Specialists shows Two-in-One

As shipped, the Tandy 1000 has some serious limitations, just like the Big Blue clone. There's not enough memory to run most of the software you need to run; there's no serial port; you have to keep setting the clock on power up and programs are so large, you're constantly swapping floppies.

Hard Drive Specialists were showing their cures for the MS-DOS woes — the *Two-in-One*.

The Computer Center makes discounts on disk drives

There's never quite enough storage on a new computer system. Almost every MS-DOS program written requires two disk drives just to get up and running.

The Computer Center was selling both internal and external, single- and double-sided disk drives for your favorite machine. They were also discounting almost their entire line of products through special purchases made for the show.

The Small Computer Company is a big deal

There's a lot of talk about integrated software these days, and much of what we are seeing is close to filling our needs, but there always seems to be something missing. Maybe the answer isn't with the big computer companies, but with the Small Computer Company, who showed their *filePro 16* package.

filePro 16 is an off-the-shelf menu-driven electronic filing system, database management system and application developer in a single package. It runs on all the Tandy MS-DOS machines.

Hoot your wares

Owlware was showing the deal of the show with the Owl 10,000 PC clones in kit or assembled form. This is a kit computer that fits your budget. You can pay for their labor or, if you're a real hardware hacker, you can buy the kit and save the labor costs.

The Shack's warehouse

Some of the best deals of the show were in the Radio Shack booth. Terminals for less than $150, pocket computers with cassette deck and software for less than $50.

Ross Computer Services with new products

There always seems to be another expansion for MS-DOS computers, and the founder of Ross Computer Services was proud to set up their booth, showing their line of Tandy 1000 and Model 100 products.

PCM tells its own story

What would a festival be without the host? I guess we'll never know, because PCM, and her sister publications, were out in force. *VCR*, Rainbow on Tape and the whole line of Falsoft products were also at the booth.

Spectrum bats a 1000

There are a lot of MS-DOS users moving from their Color Computer up to MS-DOS. Bob Rosen of Spectrum Projects was introducing a way of getting all your files from one machine to another.

The answer lies in *CoCo Copy*, a utility that reads your TRS-80 Color Computer disks on your MS-DOS computer. You no longer need an MS-DOS computer and the associated $400 software to carry your work back and forth. Now you can use a $500 Color Computer system and a $40 text editor, and have document compatability anywhere you go.

PCM

How to Make Cents of Your MS-DOS System

By William Barden, Jr.
PCM Contributing Editor

"Here's a program I just wrote to do some simple bookkeeping. My Tandy 1000 is giving me bad answers!" my vociferous friend, Fred, announced. "Why doesn't it give me exact results?" he shouted.

"Let's take a look," I offered.

Sure enough, his listing indicated dollar and cents values of $345.56778, $34.9766 and other anomalies.

"You really should have a PRINT USING to get a rounded-off value," I suggested.

"Yeah, but why doesn't the computer give exact results? I'm not doing any division, just adds and subtracts. I realize that if you divide $100.00 by 12 months, you're going to get $8.3333. . . , but you'd think the system would know how to take care of cents automatically . . . ," his salesperson's voice, usually imbued with adrenaline from listening to a dozen "How to Close the Deal" motivational tapes, died off.

"Oh yeah, here's another thing I tried," he said, renewed. "This short program gives the wrong results, too." What Fred had showed me is Listing 1.

Listing 1

```
100 FOR I = 1 TO 10 STEP .1
110 PRINT I
120 NEXT I
```

A portion of the display for the program is shown here:

```
 7.999995
 8.099995
 8.199995
 8.299996
```

"Why don't I get values of 7.9, 8.0, 8.1 and so forth — look at all the digits!"

"Fred, I've got the answers for you, if you're interested — but it'll mean digging into the internals of BASIC. Are you game?"

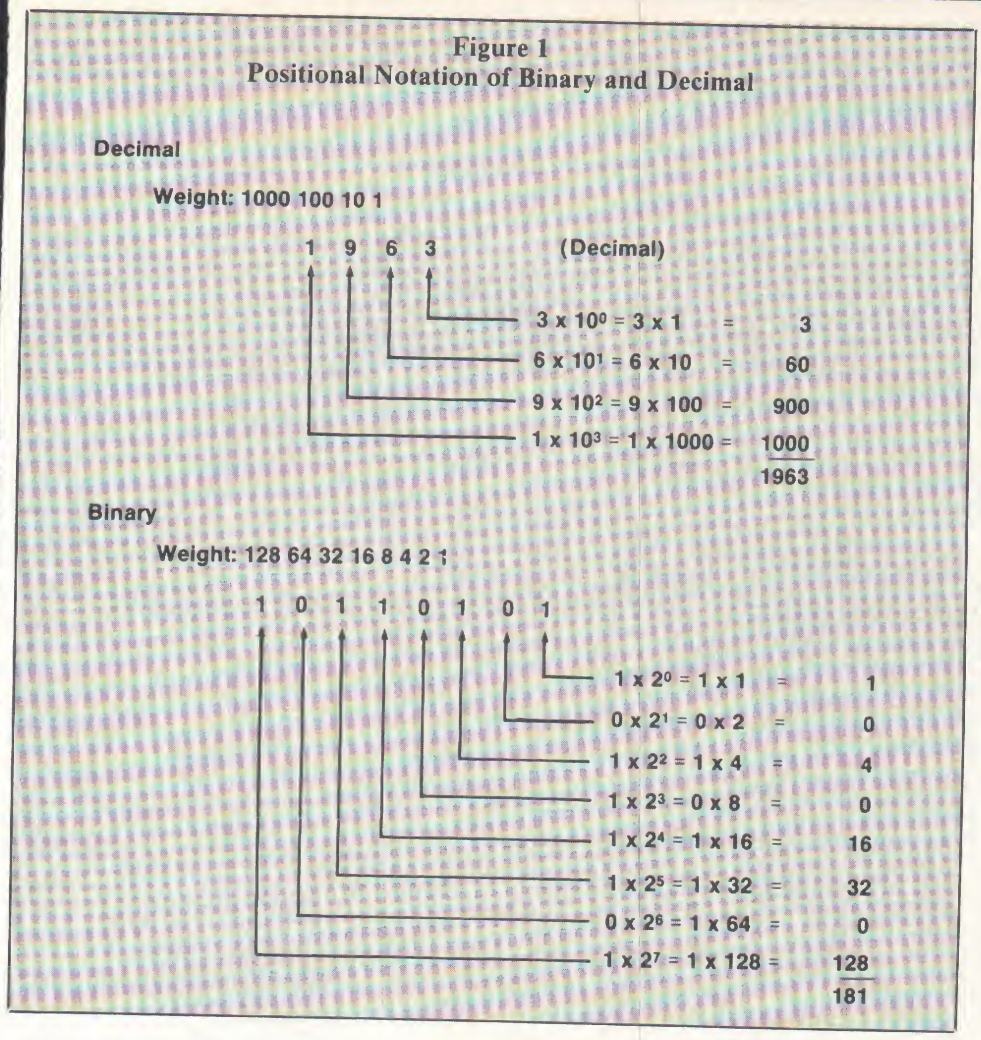
"Sure, I'm all ears," he said.

Back to Binary

"First a little binary . . . " Fred rolled his eyes.

William Barden, Jr. is a master communicator in a field in which he is one of the few recognized experts — microcomputers. A prolific author of more than 27 books on computers and computer programming, Bill also has authored several instructional software projects for Tandy/Radio Shack.

Figure 1
Positional Notation of Binary and Decimal



"I'll just go through this quickly as I'm sure you've seen it before. Everything in the Tandy 1000, 1200, 2000, 3000 and all other Tandy machines, and all *other* current computers is ultimately represented in binary numbers. Binary numbers have only two digits, zero and one. Because this is an on/off condition, the two binary states can be represented by an electrical charge or no charge, a switch that is on or off, or a magnetized spot on a diskette or no magnetized spot. I'm simplifying here a little bit, but you get the idea. 'Til someone invents a cheap, reliable electronic device that can hold 10 states to represent zero through nine, we're stuck with binary."

"Just as decimal representation represents the decimal digits' positions, a binary number represents the binary digits' positions. Engineers and programmers got tired of using the term "binary digit" about three seconds after the first digital computer was designed, and so you'll always hear the word 'bits' used instead."

"The decimal number 1,963 really represents one times 10 to the third power, nine times 10 to the second power, six times 10 to the first power and three times 10 to the zero power."

"Zero power — what's that?" Fred interjected.

"Just remember that any number to the zero power is one — 10 to the zero power is one, two to the zero power is one and so forth."

"Binary numbers use the same *positional notation* scheme as decimal, only here the powers are powers of two. The right-most position is two to the zero power (one) followed by two to the first power (two), two to the second power (four), two to the third power (eight) and so forth. Here's what I mean," I said, as I sketched a diagram (Figure 1).

"You can make the binary number as big as you want by just adding more binary digits to the left, just as you do in decimal numbers. Each digit is another power of two. Here's some equivalent binary and decimal numbers," I said, and jotted the numbers down.

Integer Numbers in BASIC

"Unfortunately though, BASIC and other languages can't store an unlimited number of bits. Oh, they *could* make each variable a kind of variable length operand, but it's a lot more convenient to have a fixed number of bits for variables. Since the hardware — the 8088 (or 80186 or 80286) microprocessor and memory — are oriented toward *byte* operations — eight bits per byte — BASIC uses multiples of bytes to store variables. The simplest variable BASIC uses is an *integer* variable. Integer variables are 16 bits, or two bytes long. Actually, it's easy to see how BASIC stores integers. Take a look at this program." (See Listing 2.)

"The VARPTR function points to the location of the specified variable. Here we're using A% as an integer variable. The percent size (%) suffix says that the variable is integer, and not the standard "single-precision" default variable type. PEEK lets us read what's in the variable location. As each integer variable is two bytes long, we're doing two PEEKs.

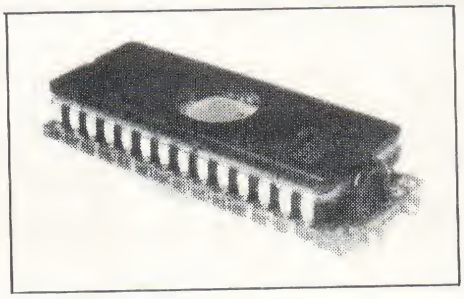
"Here's one strange thing, though, not just peculiar to the Tandy MS-DOS

Listing 2

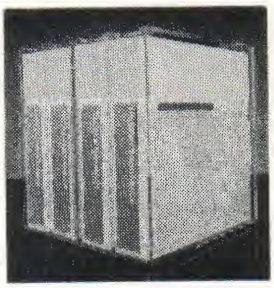
```
100 B% = 0
110 INPUT A%
120 B% =  PEEK(VARPTR(A%)+1)
130 FOR I = 7 TO 0 STEP -1
140 IF (2^I AND B%)  <> 0 THEN PRINT "1"; ELSE PRINT "0";
150 NEXT I
160 B% =  PEEK(VARPTR(A%))
170 FOR I = 7 TO 0 STEP -1
180 IF (2^I AND B%)  <> 0 THEN PRINT "1"; ELSE PRINT "0";
190 NEXT I
200 PRINT
210 GOTO 110
```

We would like to take this opportunity to introduce ourselves. Although **Ross Computer Services** is a new name to the pages of this magazine, we are *NOT* new to computers. We have been suppling computer dealers with many of the products they sell for several years. We have decided to end that relationship, and to sell directly to you, the end user.

ROMDSK. 8K or 32K CUSTOM expansion rom for the TRS-80 Model 100. Custom programmed with YOUR programs.
8K . . . . . **$49.95** 32K **$69.95**
Call or Write **ROSS COMPUTER SERVICES**
For Ordering Information



INNOVATIVE Software™



The SMART Software System, from Innovative Softwear.

The SMART DATA MANAGER	**$495.00**
The SMART WORD PROCESSOR	**$295.00**
The SMART SPREADSHEET (With Graphics)	**$395.00**
The SMART SPELLCHECKER	**$139.00**
The SMART SYSTEM*	**$895.00**

*The SMART Software System includes: The Smart Data Manager, The Smart Spreadsheet with Graphics, and The Smart Word Processor, and can only be sold as a system, not as individual products. The smart Software System includes built in asynchronous communications and personal time management programs.

BORLAND INTERNATIONAL

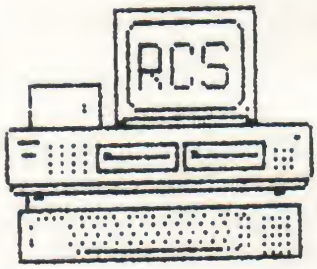
Turbo Tutor	**$ 34.95**	Superkey	**$ 69.96**
Turbo Pascal 3.0	**$ 69.95**	SideKick (copy-protected)	**$ 54.95**
Turbo Pascal with 8087	**$109.90**	SideKick (NOT copy-protected)	**$ 84.95**
Turbo Pascal with BCD	**$109.90**	Traveling SideKick (when available)	**$ 69.95***
Turbo Pascal 3.0 with both 8087 & BDC	**$124.95**	Traveling SideKick with SideKick (NCP)	**$125.00**
Turbo Database Toolbox	**$ 54.95**	Turbo Lightning	**$ 99.95**
Turbo Graphix Toolbox	**$ 54.95**	Reflex: The Analyst	**$ 99.95**
Turbo Editor Toolbox	**$ 69.95**		
Turbo Gameworks	**$ 69.95**		

*Traveling SideKick should be available by the time you read this ad.

MERCURY 300/ 1200 MODEM

Mercury 300/1200 auto-answer/auto-dial, HAYES compatible MODEM. Uses the industry standard AT command set.
SPECIAL **$199.95**

Ross Computer Services is a division of a company that has been in business for over 54 years. We are not likely to fold our tents, and steal into the night. We will sell only the *highest quality hardware and software.* Though our initial offering is limited, we will be constantly adding to our product line, again, adding only products of the highest quality.



ROSS COMPUTER SERVICES

(816) 363-2545

VISA* MasterCard

Hours: 8:30 to 5:00
After 5:00 you will reach our order line BBS

1800 W. 91st Place • Kansas City, MO 64114

systems, but to all 8088-type microcomputers. The two bytes are stored in memory in *reverse order* — the 'least significant' first, followed by the 'most significant.' The most significant byte holds larger powers of two. This program first converts one byte to binary and then the other for a display of the 16 bits.

"Here are some sample displays of integer values."

```
? 23
ØØØØØØØØØØØØ1Ø111
?1ØØ
ØØØØØØØØØ11ØØ1ØØ
?32767
Ø111111111111111
? 32768
Overflow in 1ØØ
```

Notice that at a decimal value of 32,767, the corresponding binary is 011111111111111 in 16 bits. The next bit (the zero) represents the 32,768th position (two to the power of 15). However, when 32,768 (equivalent to 1000000000000000 in binary) is entered, "overflow" results. The reason for this is that BASIC allows positive integer values of only zero (0000000000000000) through +32,767 (011111111111111). If BASIC doesn't see a sign, it assumes a positive number. The 16th bit is used as a "sign" bit. For the values above, the sign bit is a zero for positive. Let's see what happens when a negative integer value is entered:

```
? -1
1111111111111111
```

Here, the 16th bit was used. But if it represents a sign bit of one, or negative, why is the remainder of the number all ones instead of 00000000000001?

Two's Complement

Microsoft *could* have used a "sign/magnitude" representation for negative integer numbers in BASIC, where a –1 would be 1000000000000001, but there are problems with the scheme. For one thing, there would be two zeros 0000000000000000 and 1000000000000000. Instead, a scheme called "two's complement" is used. In this method of representing negative numbers, the positive representation of the number is inverted — all ones are changed to zeros and all zeros to ones — and then one is added to the result. The number –23, for example, would be derived from +23 as follows. I did some more sketching and come up with what's in Figure 3.

Figure 2
Sample Binary Numbers

Binary	Decimal	Binary Weights
1	1 =	1
10	2 =	2
111	7 =	4 + 2 + 1
1010	10 =	8 + 2
11100	28 =	16 + 8 + 4
101011	43 =	32 + 8 + 2 + 1
1101110	110 =	64 + 32 + 8 + 4 + 2
10110001	177 =	128 + 32 + 16 + 1
101011111	351 =	256 + 64 + 16 + 8 + 4 + 2 + 1
1011011011	731 =	512 + 128 + 64 + 16 + 8 + 2 + 1
10110000111	1415 =	1024 + 256 + 128 + 4 + 2 + 1

Figure 3

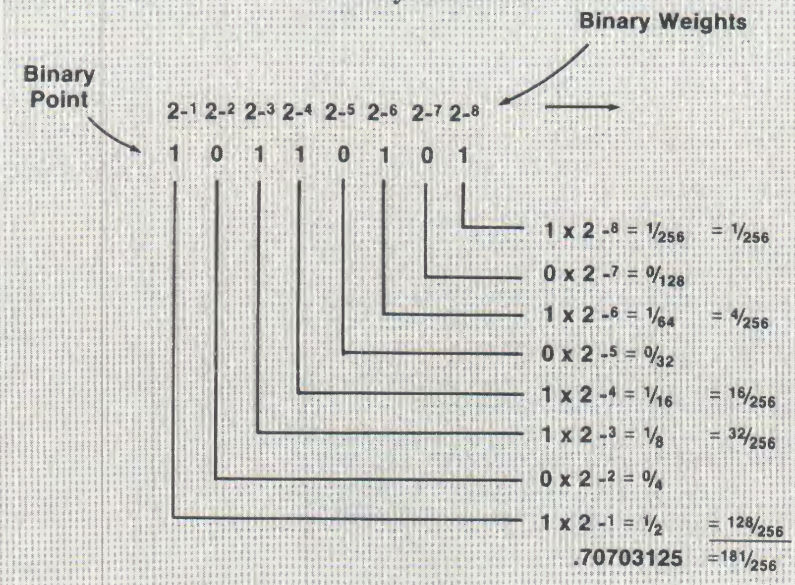
```
0000000000010111  +23
1111111111101000  all zeros to ones
                  and all ones to
                  zeros
              +1  add one
----------------
1111111111101001  -23 in two's com-
                  plement
```

"That's a complicated, crazy scheme," Fred ventured.

"Crazy? Well, certainly harder to work with from a programmer's standpoint, but commonplace in computers." Here are some more examples:

```
? -2
1111111111111111Ø
? -1ØØ
1111111111ØØ111ØØ
? -32767
1ØØØØØØØØØØØØØØ1
? -32768
1ØØØØØØØØØØØØØØØ
```

Figure 4
Binary Fractions



Note that in this scheme the sign bit for negative numbers is always a one. Also, the maximum negative number that can be held is –32,768, one more in magnitude than +32,767.

Fractions in Binary

Integer numbers in binary are not too hard to understand, but what about fractions? Any number base can represent fractions as well as integer numbers, and that applies to binary as well. The powers of two in a fraction, though, are *reciprocal powers* of two — 2^{-1}, 2^{-2}, 2^{-3}, and so forth. Another way of expressing reciprocals is $1/2^1$, $1/2^1$, $1/2^3$ . . . Whereas integer bit positions represent weights of 1, 2, 4, 8, 16 and so forth, fractional bit positions represent ½, ¼, ⅛, $^1/_{16}$, $^1/_{32}$ and other reciprocal powers of two. The scheme works like this (Figure 4).

Single-Precision Numbers

Integer numbers can be used for many operations where you don't need large magnitudes. The *default* numeric variable type in BASIC, however, is *single-precision*. It's what you get when a variable name is specified without a suffix such as '%' (integer) or '#' (double-precision).

The idea behind single-precision is to allow you to hold a wide range of both

Listing 3

```
100 B%=0
110 INPUT A
120 B% = PEEK(VARPTR(A)+3)
130 FOR I = 7 TO 0 STEP -1
140 IF (2^I AND B%) <> 0 THEN PRINT "1"; ELSE PRINT "0";
150 NEXT I
160 B% = PEEK(VARPTR(A)+2)
170 FOR I = 7 TO 0 STEP -1
180 IF (2^I AND B%) <> 0 THEN PRINT "1"; ELSE PRINT "0";
190 NEXT I
200 PRINT
210 B% = PEEK(VARPTR(A)+1)
220 FOR I = 7 TO 0 STEP -1
230 IF (2^I AND B%) <> 0 THEN PRINT "1"; ELSE PRINT "0";
240 NEXT I
250 B% = PEEK(VARPTR(A))
260 FOR I = 7 TO 0 STEP -1
270 IF (2^I AND B%) <> 0 THEN PRINT "1"; ELSE PRINT "0";
280 NEXT I
290 PRINT
300 GOTO 100
```

integer and mixed numbers that contain an integer and fraction. Also, since only about 4½ decimal digits can be specified in integer, single-precision extends the *precision* to seven decimal digits.

As in integer, we can use the VARPTR to investigate the format of single-precision numbers. See Listing 3.

The code in Listing 3 does pretty much the same as the previous code for investigating integers — it prints out four bytes instead of two bytes, however. The bytes are also rearranged somewhat, as shown in Figure 5.

The four bytes displayed represent a single-precision number and its a little more complicated than integer format, as it uses binary fractions and also an "exponent."

Scientific Notation

The single-precision number is very similar to scientific notation used in chemistry, physics, mathematics and other sciences. Scientific notation is used primarily to get rid of a lot of leading or trailing zeros and to make computations easier. It was especially valuable in the days before computers when there were a lot of hand computations done.

As an example, consider the number of inches to the moon. The moon is roughly 238,000 miles from the earth, so the number of inches is 238,000 miles x 5280 feet/mile x 12 inches/ft = 15,079,680,000 inches.

Is there an easier way to express this number? The way used in scientific notation is to "normalize" the number to a single integer digit and a fraction by using a power of 10 exponent: 15,079,680,000. x 100 = 1.5079680 x 1010.

To get the power of 10, just count off decimal places and use the count as the exponent. For every place counted to the left, add one to the exponent; the exponent starts with zero, as 10 to the zero power is one.

Here's another example. Consider how large a square inch is in acres. One acre/(43,560 square feet x 144 square inches/square foot) = 0.0000001531 acres/square inch.

To get the scientific notation of this number, count off decimal digit positions to the right of the decimal point and subtract one from power of ten exponent, starting with zero: $0.0000001531 \times 100 = 1.531 \times 10^{-7}$.

When large and small numbers are represented in scientific notation, they can be easily manipulated — multiplying two numbers involves multiplying the fractional part and adding exponents, while dividing two numbers involves dividing the fractional part and subtracting exponents: $1.5 \times 10^5 / 2.5 \times 10^2 = 1.5 / 2.5 \times 10^{(5-2)} = 0.6 \times 10^3$.

BASIC allows you to use a form of scientific notation for single-precision variables with the 'E' format. The number 15,079,680,000 = 1.5079680×10^{10} is expressed in BASIC as `1.5E10`, or similar forms, such as `15E+9`. The number $0.0000001595 = 1.594 \times 10^{-7}$ is expressed as `1.594E-7`. In these forms, the "10," or *base* of the power, is understood.

Single-Precision Floating Point versus Scientific Notation

The four bytes displayed in the program I just showed you are also broken down into a mixed number and an exponent. In BASIC's single-precision format, however, the exponent is a power of *two* instead of 10. The mixed number is actually a fraction with the *binary point* to the extreme left (see Figure 5).

Let's take an example. Suppose we wanted to represent the decimal number 213.75. From conversion tables or conversion methods, you can convert 213 in decimal to 11010101 binary. The fractional part of the number is equivalent to binary 0.11, equal to ½ + ¼ or ¾ or .75. The complete binary number with binary point is 11010101.11.

The next step is to convert to the form of scientific notation used in single-precision numbers in BASIC, with a power of two and no integer portion. We can do this by moving the binary point to the right and simultaneously adding one to a power of two:

```
11010101.11 x 2⁰
1101010.111 x 2¹
110101.0111 x 2²
11010.10111 x 2³
1101.010111 x 2⁴
110.1010111 x 2⁵
11.01010111 x 2⁶
1.101010111 x 2⁷
.1101010111 x 2⁸
```

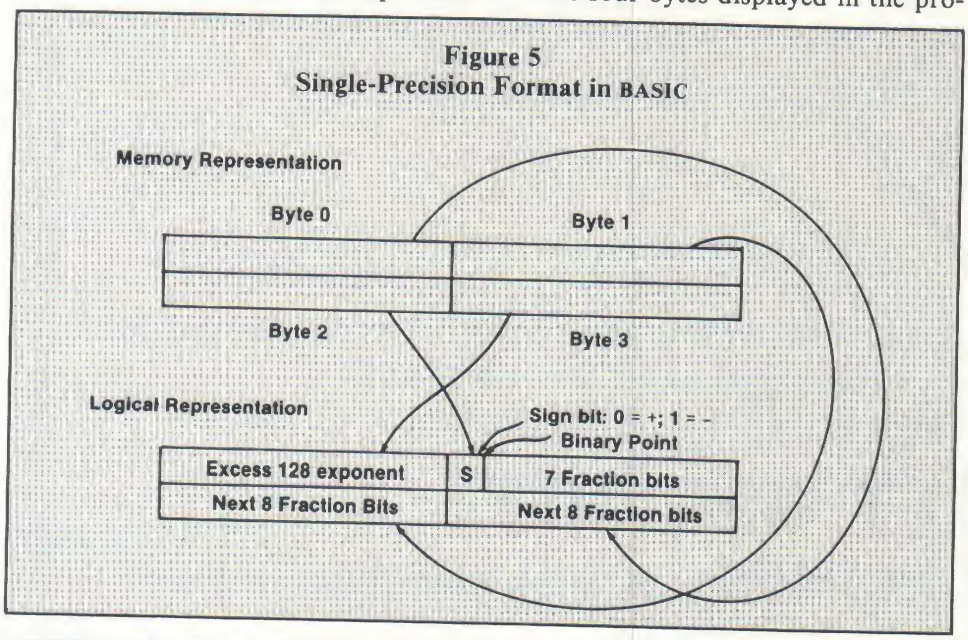
Entering 213.75 into the program above, we get:

```
? 213.75
10001000010101101
11000000000000000
```

Looking at the format (Figure 5), we can see that the sign is zero, or positive, the fractional part is .10101 01110000000000000 and that the exponent part — the power of two — is 10001000. There are two problems here — where's the missing one in the first bit of the fraction and why is the exponent 10001000 instead of 00001000?

The first bit of the fraction is missing on purpose. Microsoft uses a scheme that says, in effect, the first bit to the right of the binary point will *always* be a one bit, so why not simply discard it? The advantage of discarding the first bit is that it squeezes an extra bit of *pre-*

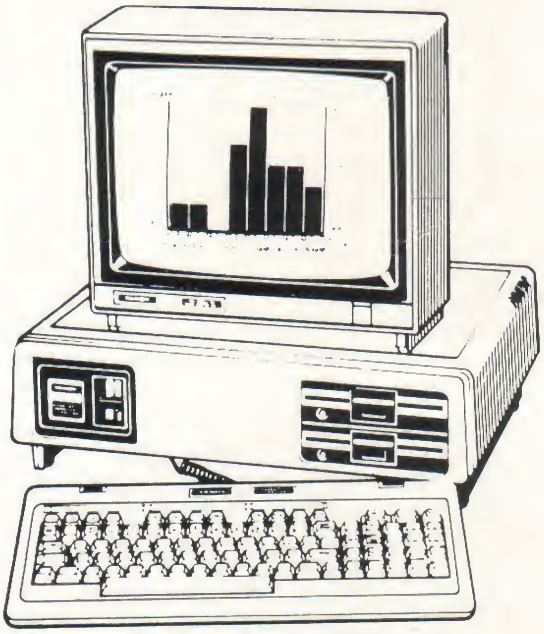
Figure 5
Single-Precision Format in BASIC



NOCONA

ELECTRONICS

TRS-80 Model 2000 by Tandy®



TRS-80® Model 100 by Radio Shack

MS-DOS Based

Computer System

256K MOD2000 W/MONO MONITOR	1319.00
256K MOD2000 W/COLOR MONITOR	1629.00
256K MOD 1200 HD 2 DR.	1499.00
128K MOD 1000 W/MONO 1 DR.	759.00
128K MOD 1000 W/COLOR 2 DR.	1099.00
256K MOD 1000 W/HARD DISK	1439.00
24K MOD 100 PORT	399.00
24K MOD 200 PORT	679.00
64K MOD 4D 2 DR	839.00
512K MOD 6000 2DR.	1849.00
512K MOD 6000 w/15MEG HD	2579.00
15 MEG HARD DISK	1119.00
35 MEG HARD DISK	2059.00
DMP 105 PRINTER	145.00
DMP 2200 PRINTER	1149.00
DMP 130 PRINTER	249.00
DMP 430 PRINTER	585.00
DMP 2100P PRINTER	1019.00
CGP 220 COLOR PRINTER	449.00
DWP 510 PRINTER	999.00
TRACTOR DWP 210	99.00
TRACTOR DWP II/510	159.00
TRACTOR FEED DMP2100P	99.00
PARALLEL PRINTER SWITCH	96.00
PARALLEL PRINTER CONTROLLER	179.00
PARALLEL PRINTER CABLE	29.00
DISK VIDEO INTERFACE	399.00
32K MOD 600 PORTABLE	1149.00
512K MOD 3000	1819.00

100% RS COMPONENTS NO FOREIGN DRIVES OR MEMORY FULL WARRANTY
ALL RS SOFTWARE 20% OFF CATALOG PRICE
CASHIERS CHECK OR MONEY ORDER MUST ACCOMPANY ALL ORDERS

(817) 825-4027

NOCONA ELECTRONICS • BOX 593 • NOCONA, TX 76255

cision out of the fractional part. Putting in the missing one bit, which is always there in any single-precision number, therefore, gives .1101010111 00000000000000, the proper fraction.

But what about the exponent? Why is it 10001000 instead of 0001000 or eight, the proper power for the example of 213.75? The reason is that Microsoft uses a form of exponent representation called "excess 128." In this format, 128, or binary 10000000, is added to the eight bits of the exponent. To reconvert, we have to subtract the 128 once again. In this case (10001000 - 10000000) equals 00001000, the proper exponent value. Actually, just flipping the leading bit will do the subtraction. Since both positive *and* negative powers of two must be represented in the exponent, the excess 128 scheme provides a range of positive and negative exponents that look something like Figure 6.

Figure 6

```
11111111 = 01111111 = 2+127
11111110 = 01111110 = 2+126
.
.
10000011 = 00000011 = 2+3
10000010 = 00000010 = 2+2
10000001 = 00000001 = 2+1
10000000 = 00000000 = 2°
01111111 = 11111111 = 2-1
01111110 = 11111110 = 2-2
01111101 = 11111101 = 2-3
.
.
00000001 = 10000001 = 2-127
00000000 = 10000000 = 2-128
```

Trying the negative of 213.75, we get:

```
? -213.75
1000100011010101
1100000000000000
```

This example is identical to the first, except that the sign bit represents a negative *fraction*. Note that in the case of single-precision numbers, two's complete is *not* used — it's the sign/magnitude form we discussed before.

The examples so far have been greater than zero. Let's try an example in which the exponent is negative (see Figure 7).

Figure 7

```
0.00146484375
decimal = 0.00000000011
binary

0.00000000011      x 2°
0.0000000011       x 2-1
0.000000011        x 2-2
0.00000011         x 2-3
0.0000011          x 2-4
0.000011           x 2-5
0.00011            x 2-6
0.0011             x 2-7
0.011              x 2-8
0.11               x 2-9

? 0.00146484375
01110111100000
00000000000000
```

The sign bit here is zero, which is correct. The fraction is .X1000000 or .11000000000000000000000 with the missing sign bit added. The exponent is 01110111. Flipping the sign bit (adding 10000000), we get 11110111. This is a two's complement form of the exponent, so we'll have to convert (see Figure 8).

Figure 8

```
11110111  Original
00001000  After changing all ones to
          zeros and all zeros to ones
      +1  Adding one
--------
00001001  Final exponent repre-
          sents 2-9
```

The Range and Precision of Single-Precision

Integer numbers can represent a *range* of values from -32,768 through +32,767 with about 4½ decimal digits of precision. Single-precision numbers, though, have a much greater range because of the powers of two represented. The maximum power of two is +127, the minimum -128. The value of 2^{127} is about 1.7×10^{38} and the value of 2^{-128} is about 3.4×10^{-38}, making the range of single-precision numbers about 10^{-38} through 10^{+38}.

The *precision* of single-precision numbers can be found by looking at the size of the fraction. With the missing one bit to the right of the binary point, it's 24 bits. Numbers from 000000000000000000000000 through 111111111111111111111111 can be held in 24 bits, representing zero through 16,777,215 decimal. The precision of single-precision numbers is therefore a little over seven decimal digits. That explains why entering PRINT 1/3 in BASIC results in 3.333333.

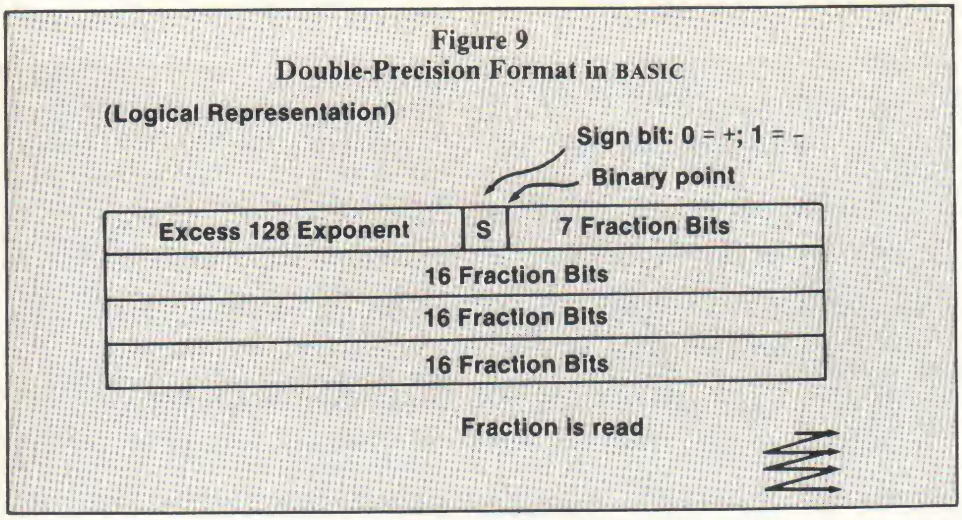
Now What about that .1 Increment?

The way this discussion first started was investigating why BASIC couldn't provide exact answers. You can see now that it provides exact answers if you limit yourself to integer values of 4½ digits. Any result produced by single-precision, though, is *not* going to be exact, unless it's a power of two to begin with. Take the original example, adding 0.1 to a count. Any fractional decimal number can be converted to binary by multiplying by two and using the integer to the left of the decimal point as the binary digit. In this case we have:

```
      .1
    x 2
0←  .2
    x 2
0←  .4
    x 2
0←  .8
    x 2
1←  .6
    x 2
1←  .2
    x 2
0←  .4
    x 2
0←  .8
    x 2
1←  .6
    x 2
1←  .2
    x 2
0←  .4
    x 2
```

From the above, it looks like 0.1 in binary is going to be a repeating series of the form 0.000110011001100. . . This is verified when we use the single-precision program to look at an entry of 0.1:

Figure 9
Double-Precision Format in BASIC




```
? Ø.1
Ø111111Ø1Ø1ØØ11ØØ
11ØØ11ØØ11ØØ11Ø1
```

The error in this representation is one part in 2^{24}, or one part in 16,777,216. It's this error that leads to values of 3.099999 instead of 3.1. Over a long run of increments the effects of the error is even more obvious, as the error is accumulative. Running the program in Listing 4 and looking at the final value, we see 99.99904 instead of the expected 100, an error of about one part in 10,000.

Listing 4

```
1ØØ FOR I = 1 TO 1ØØ STEP .1
11Ø PRINT I
12Ø NEXT I
```

How to Deal with BASIC Errors

The errors in BASIC and other languages are not going to go away. In most cases, they're not even bothersome — it's only when you need exact accuracy that single-precision errors are important. For good customer relations with an accounting package, for example, it's nice to have the cents balanced to the penny.

One partial solution to the problem is to use *double-precision* variables instead of single-precision variables. Double-precision variables have the format shown in Figure 9.

Here the exponent part is the same, but an additional four bytes have been added for the fractional part of the number, making the total number of fractional bits 56. There'll still be an error, but it will be one place in 2^{56}, about one place in 16 decimal digits. The total number of decimal digits with 56 bits is also about 16 decimal digits. You'll be able to do a large number of calculations before any effects of accumulative error are felt. As an example, look at this increment (Listing 5).

Listing 5

```
3ØØØ I# = 1: INC# = .1
3Ø1Ø I# = I# + INC#: PRINT I#
3Ø2Ø GOTO 3Ø1Ø
```

After 4,940 iterations (repeats) of this loop, the value printed stands at 494.0000073462725 instead of 494, an error of about one place in eight decimal places. If the number represented dollars and cents, you could keep the cents accurate, provided you did rounding, as numbers such as 493.9999957575999 are equally likely — the same error, but in the opposite direction.

Rounding is easy if you use PRINT USING — it rounds automatically. If the number kept internally is 493.9999957575999, for example, doing a `100 PRINT USING "$$#,###.##;" AMT` would print `$494.00`.

In the increment example (Listing 5), you never see an amount other than one representing dollar multiples of .10 — you'd see $92.40, $92.50, $92.60, etc.

Another way to get an *absolute precision* is to use a separate variable for cents and dollar amounts. If the amounts handled are never greater than seven digits, and only addition, subtraction and multiplication are to be done, you can use single-precision for the dollar amounts. The dollar amounts will be exact for values up to $9,999,999. The cents values can be held in another single precision variable. Of course, it does make calculations a little laborious (see Listing 6).

Is there such a thing as *absolute precision*? One scheme is BCD, or binary-coded-decimal. In this method of representing numbers, four bits hold these values:

```
0000 = 0
0001 = 1
0010 = 2
0011 = 3
0100 = 4
0101 = 5
0110 = 6
0111 = 7
1000 = 8
1001 = 9
```

Combinations of bits other than these (1010, 1011, 1100, 1101, 1110 and 1111) ordinarily allowed in binary, are not allowed for BCD. A BCD number may be as long as necessary, with two decimal digits represented per byte. Additions, subtractions, multiplications and divisions are done in BCD fashion. Many early computers were BCD machines, but binary representation became the *de facto* standard years ago. BCD can be implemented in BASIC with some trouble and loss of speed. It's a better bet in assembly language, where there are special BCD instructions, or in higher languages like Borland's *Turbo* PASCAL, which offers a BCD option for MS-DOS machines.

"And so, that's about all I know on the subject of why BASIC can't represent numbers exactly. Any questions, Fred?" I asked, realizing I hadn't heard a peep out of him for several minutes.

Fred was fast asleep, with a smile on his face. Probably dreaming about killings in the time-share condominium field, I thought. "Wake up, Fred," I said.

"Mpph. Oh, guess I must have dozed off during the last part of your diatribe on single-precision. Forget it — I'll just use *Lotus* instead." □

Listing 6

```
1ØØ NEWDOL = NEWDOL + TRANSDOL        'find new dollar amt
11Ø NEWCNTS = NEWCNTS + TRANSCNTS     'find new cents amt
12Ø IF NEWCNTS > 1ØØ THEN NEWDOL =
    NEWDOL + 1: NEWCNTS = NEWCNTS -1ØØ  'bump dollars
```

PCM

A screwdriver, 10 minutes and a $25 chip are all that's needed to increase your 1000's speed by 10 to 24 percent

Supercharge your Tandy 1000

By Brian H. Alsop

What's faster than a speeding PC XT? A Tandy 1000 is — if you replace its 8088 central processor with a NEC V-20. Tests show that one can expect up to 24 percent faster calculations with the NEC CPU. For $25, it's the best bargain in speed since the $8 crystal for the PC AT. This article describes the installation and testing of the V-20 in the Tandy 1000. Actually, testing took longer than installation. Anybody with a screwdriver and two fingers can do the installation.

Purchase and Installation

My V-20 CPU was purchased from Exec-PC, P.O. Box 11191, Shorewood, WI 53211. It arrived three working days after purchase. The replacement CPU comes with complete instructions for replacing the CPU in the PC XT. The instructions apply as well to the Tandy 1000 — except the 8088 CPU is located differently and the case is removed differently. Follow Exec-PC's instructions with the following changes.

Brian Alsop is an engineer working on introducing personal computers into the engineering work environment. He is a pilot and holds an amateur radio operator's license.

Before doing anything, unplug the power cord, keyboard and other cables attached to the Tandy 1000. Wait about 30 seconds. Unscrew the two front panel screws and slide the cabinet forward.

In the Tandy 1000, the 8088 CPU is located in a front-to-back line with the middle expansion slot. It is clearly marked "8088" as well as with some Tandy numbers.

Follow the 8088 removal instructions provided. It is particularly important to bend the pins of the NEC V-20 inward so that it will fit in the socket. ICs are normally shipped with their pins angled outward. The instructions suggest two ways to bend the pins to fit. A better way is to set the IC on a flat surface with one row of pins on the surface. Gently roll the IC to bend the pins in the proper direction. This keeps all pins in a line. Repeat the process for the other row of pins. Compare them to the removed 8088. When the two show the same pin spacing, plug the V-20 in as described. The notch should be toward the front panel. The circuit board has the notch position indicated as well.

Carefully inspect the installation for pins that are bent or not inserted. Push the IC down as far as it can go. Mine first went in and then popped in even further with a little more force. Slide on the cover and test the installation.

Grow up!

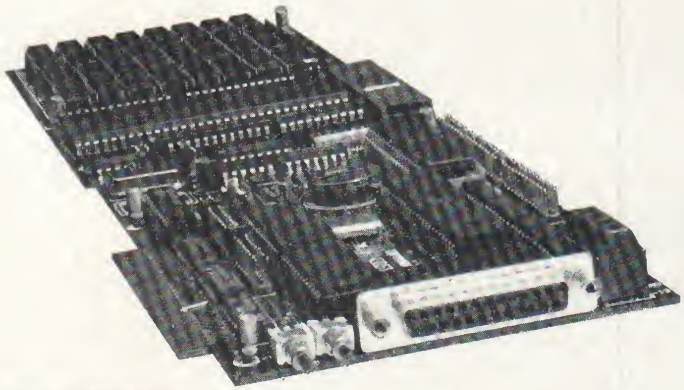
Wondering where to grow?

Why not get IBM compatibility with Tandy's 1200, or the new 3000, and take your OS-9 programs with you?

TLM's PC68K puts the 68000/68010 processor and Microware's OS-9 together in any IBM, or compatible, PC/XT/AT system, providing the best of both worlds with fully concurrent PC/68000 operation.

- Powerful languages such as BASIC09, PASCAL and C
- DYNACALC for spread sheet processing
- The power of the 68000/68010
- The price/performance and flexibility of the Tandy 1200 and 3000

TLM's PC68K . . . A great family to grow up with!



CLOCK SPEED
8/10/12.5 MHZ
Without wait states

HARDWARE
2 Parallel Ports
3-9 Serial Ports
68881 Floating point co-processor (optional)
Memory mapped dual port bus for high speed DMA transfers
Real time clock with battery backup
Up to 16 boards can run concurrently from one AT/XT/PC bus

MEMORY
1-2 MB OF RAM
(1 MB on board standard)
8K-32K of ROM
2K-8K of battery backed RAM

MULTI-USER
3-9 users (3 standard)
High speed "Tick" timer with 10ms accuracy for efficient multi-user time slicing.

OPERATING SYSTEMS
OS-9 (Multi-user UNIX look-alike)
C/PM-68K (popular single user OS)
Both support concurrent access to PC DOS

SOFTWARE
Software selectable OS environment
IBM Color/Mono Graphics Support
Local and Global disk caching for maximum speed and efficient data transfer

COMMUNICATIONS
Powerful asynchronous/synchronous data communications capability
Bisynchronous SNA 3270, X.25, etc.

MISCELLANEOUS FEATURES
Remote and Local reset (hardware or software selectable)
Remote and Local Non-maskable Interrupts (NMI)
Stand alone hardware capability (without AT/XT/PC support)

TLM Systems

West: 4704 W. Jennifer, Suite 105, Fresno, CA 93711 209/276-2345 / East: 67 Grandview Avenue, Pleasantville, NY 10570 914/747-1450

Testing and Using It

Now for the fun. Plug in the Tandy 1000, keyboard and monitor. Turn the monitor on. After it's warmed up, turn the Tandy 1000 on. The amount of installed memory should immediately be displayed. If it is not, quickly turn off the Tandy 1000 — you made some error that might or might not cost $25 to fix; inspect the installation for bent or uninserted pins. If the proper amount of memory is displayed, you've been successful.

Tighten the two front screws and connect all peripherals. The V-20 is advertised to be 100 percent compatible with 8088 software. It gets its increase in speed by using fewer instructions to perform memory reads and writes as well as some other operations.

I've tested this unit with BASIC, FORTRAN and compiled BASIC programs created before V-20 installation. Several commercial programs were also tested successfully.

Table 1 shows the percentage improvements obtained in BASIC with the V-20 installed. Pure math operations such as addition and multiplication were sped up 24 percent. All input-output operations to disk were no faster. The few FORTRAN problems tested yielded a 10 to 15 percent improvement. The word processor program used seemed to display new pages appreciably faster. Unfortunately, the long wait for the system to boot up is no faster. Expect at least a 25 percent improvement in input-output to a virtual disk.

Note that the Tandy 1000 is now at least 15 percent faster than the stock PC XT.

Is it worth it? I'd say yes — if you work a lot in BASIC. I don't know about FORTRAN or assembly language. If a job takes seconds, the benefit is probably not worth the cost. If you run jobs that take tens of minutes (a big spreadsheet) then even a 10 percent savings is worth it. You decide.

Now for the bonus. Exec-PC allows you 10 days free access to over 3,000 free programs on their bulletin board. You certainly can't lose with this speed-up kit! □

Table 1. Computer Speed Comparisons

Operation	Tandy 1000	Tandy 1000/V20	PC XT	PC AT
Subtract	41	33	38	15
Add	41	33	38	15
Multiply	43	31	40	15
Divide	43	31	49	19
I/O Floppy	27	27	25	14
I/O Hard Disk	19	19	20	8

Notes: 1) Times are in seconds
2) 10,000 math operations
3) I/O problem read/write of a file containing 1,000 lines of "HI THERE"

PCM

With any collection of public domain programs, the name of the game is . . .

Share and Share Alike

Kevin Nickols
MS-DOS SIG Manager

Of all the features of the Delphi MS-DOS SIG, it's the database that can prove the most valuable to the members. This is where you can share the programs you write with other users, and where you can download programs yourself — everything from short utilities to some of the finest full-featured terminal programs available anywhere.

But the MS-DOS SIG database is a collection of programs uploaded and submitted by the members. In other words, it takes a lot of member participation to build a really good collection. The more all the members are willing to participate, the better the database collection will become.

For this reason, let's look at the process of uploading a file to your Workspace in the SIG, the first step in sharing programs with other Tandy users.

The first thing you have to do is get the program transferred, over the telephone line, to the Delphi host. Now, a high-speed data transfer can be an intimidating, mind-boggling thing if you try to put too much thought into it. But in practice, it couldn't be more simple.

Any program uploading to the SIG must be done in the area known as your Workspace. At the MS-DOS SIG prompt, type DA to enter into the database and then select one of the topics by typing in the first few letters. (Remember that it really doesn't matter which topic is selected; you can still submit a program to any of the other topics.) From here you can enter the Workspace by typing WO.

Inside the Workspace, you will find that it is nothing more than a file storage area, just like a hard or floppy disk. Many of the MS-DOS file commands you're already familiar with function just the same as they do on your computer. You can rename a file with REN, copy a file with COPY, get a directory with DIR, and the DOS question mark (?) and asterisk (*) wild cards function the same, too.

There are actually more ways than one to upload a file, depending on what type of file it is, what protocols your own terminal program supports, and even personal preference. If you want to upload a pure ASCII text file or BASIC program saved in ASCII, you can upload it as an ASCII transfer. Just type UP filename, then respond with "no" when the system asks whether you want a line feed sent after every line. Begin the ASCII transfer on your terminal and you will see all of the characters start flowing across the screen as if you were typing them in yourself. When the file has finished, send a CONTROL-Z to close the file.

Now the ASCII transfer is fine in many cases, particularly for short messages, but there is no error-checking going on during the transfer. If your telephone connection is noisy, you can wind up with characters in the receiving file that shouldn't be there. So you can see, for something critical like a BASIC program, this is not a very reliable method of transfer even though it is ASCII.

Enter the error-checking protocols, Xmodem and Kermit (not the only two, by any means, but two of the most popular and the only ones currently supported by Delphi). When transferring binary files, it is *necessary* to use one of these protocols, but you should also go this route when transferring ASCII files if an error-free transfer is imperative. Under these protocols, your computer quickly sends a block of data, then communicates with the host computer to see if the block received is exactly like the one sent. If not, it resends the same block again and keeps trying until it gets it right or the transfer is aborted.

To begin an Xmodem upload, enter your Workspace and type XUP filename. When asked whether it is a text or binary file, answer appropriately. In a couple of seconds, the system will tell you "OK, send." From here, it's up to you to begin the transfer on your computer. When it has begun properly, your terminal program will probably display information about each block as it is transferred so you know everything is moving along successfully.

For a Kermit upload to your Workspace, type KUP. Notice that it is not necessary to specify the filename. You can if you want to, but Kermit transfers the proper filename, along with other information, in the file header. Here, too, the system asks whether you will be sending only text files; answer yes or no and begin the transfer on your machine.

Now, you may have noticed the word "only" above and wondered about it. This is because, with Kermit, you can do "batch" transfers as well as single files. For instance, you can upload *.BAS and transfer every BASIC program that happens to be on the disk in your default drive at once. The only catch to this is that you cannot mix ASCII and binary files in the same batch transfer.

OK, so the command to begin an ASCII transfer is UP, an Xmodem transfer is XUP and a Kermit transfer is KUP. Simple enough, right? Just enter the MS-DOS SIG, go into your Workspace and upload a short test file using each of the protocols. Next month we'll look at some of the things you can do with a file after it has been uploaded. PCM

HELPFUL hints

This programming tip comes from Barry Erick of Dallas, Pennsylvania. It is a method for turning the caps lock on and off from within a BASIC program running on a Tandy 1000 or 1200:

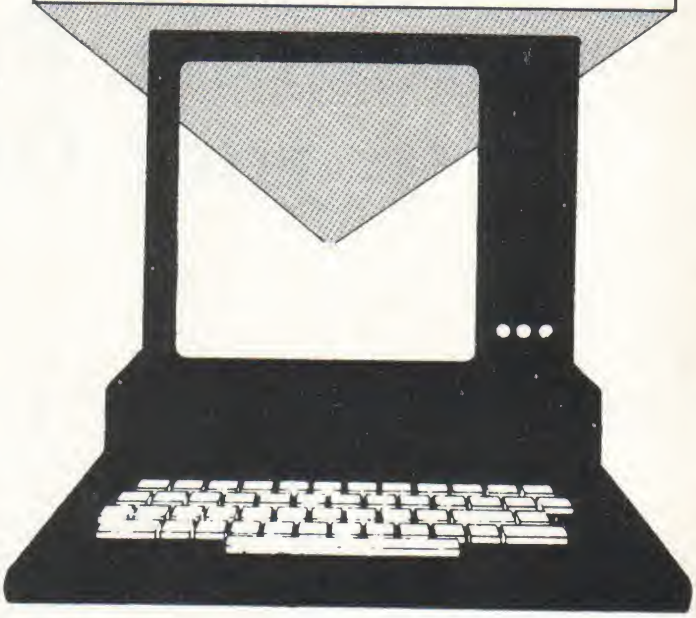
```
DEF SEG=Ø: POKE 1Ø47,PEEK(1Ø47)
 OR 64  'Caps Lock On

DEF SEG=Ø: POKE 1Ø47,PEEK(1Ø47)
 AND 32 'Caps Lock Off
```

Send your hints and tips to PCM Helpful Hints, P.O. Box 385, Prospect, KY 40059. Selected winners will receive $10 upon publication.

Correction for Graphics Screen Dump

There was a line omitted from the listing of *Graphics Screen Dump*, by Peter T. Lee (March, 1986). The missing line and the lines directly above and below it are listed here.

```
ØD2Ø:Ø15D Ø1D8          ADD     AX,BX
ØD2Ø:Ø15F 89C1          MOV     CX,AX
ØD2Ø:Ø161 2E            CS:
```



Play Writer — Write, Edit, Illustrate, Print and Bind your Own Book

Software 1000/1200/2000/3000

"Write, Edit, Illustrate, Print, and Bind Your Own Book" — this quote is from the cover of *Mystery*, one of the *Play Writer* series. *Play Writer* comes with different versions for *Adventures In Space*, *Tales Of Me*, *Mystery* and *Castles & Creatures*. All versions, except one for Apple only, are available for IBM PC and PC*jr*, Commodore 64 and Apple. This review is of *Mystery* for the IBM PC as used on a Tandy 1000 with 256K. As usual, the Tandy worked fine with a product designed for the IBM PC.

Play Writer does indeed help to write, edit, illustrate, print and bind your own book. The software guides you through the writing of a five chapter story with some of the writing being provided by *Play Writer* and the rest provided by you, with helpful prompts and choices as a guide. Simple editing features are included so rough drafts of your book can be reviewed and modified as desired. Four full-page color illustrations and 30 colorful and appropriately selected stickers provide for the illustrations in your book. Of course, you can use your own stickers or artwork if you wish. The final version of your book is printed on specially sized paper provided. The provided colorful hardcover is used to easily bind your finished book with the special adhesive first and last pages provided.

Play Writer comes with one disk which can be backed up, but the backup copy cannot be run. The backup can only be used to restore files on the original disk. Included in the package are a small 16-page Writer's Guide of instructions, the disk, the four full-page illustrations, the four pages containing 30 stickers, the small pack of paper for printing the book, the two self-adhesive pages for binding, the hard cover and an order blank.

The instructions cover getting started for the Apple II series, the IBM PC and PC*jr* and the Commodore 64. An optional procedure for using separate disks to store your finished stories is explained. Writing stories with *Play Writer*, saving the story, using the editor, printing the stories, printing options, binding the book and troubleshooting for problems are all covered in the 16-page instructions. The instructions are brief, simple and easy to follow. The on-screen prompts are about all that's needed once you get started. There is one caution here; the supplies for book making don't come with any extras. If you mess up part of it, you are on your own for getting around the problem unless you want to order a refill pack from Woodbury Software.

The instructions for *Mystery* state that it's for ages 9 and up. Once my 10-year-old daughter got a look at the package, she was ready to immediately try it. With a minimum of help from Dad to get her started, she was totally absorbed with the process of making her first book. It took her about two and one-half to three hours to make her first story and get a draft printed on plain paper for review. Most of this time, Dad was looking over her shoulder to see how *Play Writer* was working and answering questions about how to spell different words. Later that same day, she decided to do another story when Dad wasn't available for help with the spelling. With only one or two exceptions, she managed to complete another story on her own, including the spelling.

Based on our experience, it would seem that ages 9 and up is right. *Play Writer* provides part of the story, probably more than half. Teenagers and adults will probably not use *Play Writer* much after the initial fascination with seeing how it works. There isn't enough control over all details of the book for a serious writer, but *Play Writer* isn't aimed at serious writers. However, 9- through 12-year-old children will probably be happy to produce a quantity of stories before tiring of the package. How else could I have gotten my 10-year-old to spend several hours in one day at the computer practicing creative writing? She had a short book all written in draft form in this short time frame.

The options for setup include some limited choices about printers. For my Tandy 1000 using the IBM setup, the choice was for CR/LF or CR only. That represents carriage return and line feed or carriage return only. I used the default of CR/LF which resulted in all printed output being double-spaced as was intended. The setup also includes specifying if your computer has one or two disk drives or a hard disk.

The first menu choice that appears on the screen gives you the choice of working with an old story, starting a new story, deleting stories, changing to another story disk, telling *Play Writer* about your disk drive setup or quitting. Once you start a story, you are given about three different types of prompts to complete your part of the story.

You may be given a menu of choices such as this one listed in the instructions:

> NASA wants to use a kid in space since grown-ups don't do well on long journeys. Is it because grown-ups:
> 1. get spacesickness
> 2. miss their kids too much
> 3. are too sensible to take chances
> 4. other

At this point you enter one of the numbers (1, 2, 3 or 4). If you enter '4' for other, you are given the opportunity to enter your own reason for why grown-ups don't do well in space. A series of dashes is displayed followed by underscores "(-------_____)." The various similar prompts may use just part of one line or extend for several lines. The dashes indicate the minimum amount of text that is to be entered and the full length indicates the maximum size entry that's allowed.

Using this NASA example, let me explain how the stories get a great deal of variation. If you enter '1' as your choice, the *Play Writer* generates its portion of the story a certain way. If you enter '2', *Play Writer* will generate a variation of the story which is different from choice '1', and choosing '3' gives yet another variation. Doing it yourself by selecting other, causes a fourth variation of the story. By considering that this type of choice is made repeatedly throughout the development of the story, it's easy to see how a very large number of story variations are possible. There are many combinations of choices that could be made while progressing through the book. The stickers provided are designed to fit in quite nicely with the various story variations available.

A second type of prompt you are given is simply to fill in a blank. This is commonly used so that you can give your own names to all or at least most of the characters in the story. *Play Writer* automatically remembers the names you provide for use throughout the story.

The third type of prompt asks you to complete a sentence such as describing how the victim made his fortune, what impressed you about the mansion when you first entered, or how your client appeared when first entering your office. This makes it easy to use your imagination for adding your own creative ideas to the story.

Play Writer has short pauses while the story is saved to disk or retrieved from the disk, etc. During these pauses, neat little messages are displayed. Examples: "Please wait til I get back," "Please don't shut off your computer, I've got to clean up," "It's Been a Pleasure Working With You, Julie." That's right, it asks for your name at the beginning and uses it in some of the messages.

Another feature to help hold the interest of a young writer is the plentiful use of riddles throughout the story writing session. These usually come up when there is a pause to save the story to disk, etc. They are about like this for Mystery: "OK, I'll Put That Away," (after edit changes to your story)

"Meanwhile, Try this Riddle. What did one detective say to the annoying detective? Get off my case!"

The edit options for making revisions to the story on the Tandy 1000 are: Use the left arrow to move the cursor left within the text, the right arrow for right or the down arrow for down. For some reason, the up arrow isn't allowed. You can retype over the text already entered, use the delete key to delete or insert by use of the insert key. When editing your story, a menu of the five chapters is displayed. You enter the number for the chapter and each paragraph is displayed one at a time for editing. The F1 key calls up a help screen and the F2 key restores the paragraph to its original form before you started revisions. When finished with the paragraph, the ENTER key causes the revisions to be saved.

The display of one complete paragraph at a time is handy for editing. A menu choice is given to move back one paragraph, to move forward one paragraph, to edit the paragraph currently displayed and to quit. The quit option takes you back to the menu of chapter choices which also includes a quit choice.

One slightly annoying feature here is that the chapter numbers are not printed. You must determine for yourself where one chapter ends and another begins while editing your draft story. For example, if you want to edit a certain paragraph in the middle of your draft printout, you can only guess in which chapter the paragraph is. The chapter designations are used primarily for providing handy stopping points during writing of the book. Menu prompts are provided at the end of each chapter for saving your work or continuing to the next chapter.

Play Writer comes with just enough paper and supplies to complete one properly illustrated book. There are some extra stickers left after the first book, but much less than half if your child likes to use stickers. As the instructions state, you may make as many additional books as you wish using your own graphics supplies and paper. You could print a book in final format on your paper and cut it to size by hand. Of course, you can also order refill packs for completing another book from Woodbury Software. That may be the best choice because the book is printed in a format that matches the size of the supplied stickers. You have no control over the printed format, which automatically provides blank areas on the page for graphics. The actual binding with a stapler and the provided adhesive pages goes fairly well, but you must be careful.

The finished book is 30 pages long if you include the title page, the dedication page, four illustration pages and text pages. The pages are 5½ by 8½ with a slightly larger cover. Considering that the book is double-spaced throughout and uses generous margins, the final book isn't really very long. It's a good size to make a nice project for ages 9 through 12.

I recommend *Play Writer* if you wish to promote creative writing by children of the proper age. It can even generate a budding interest in creative writing for your child.

(Woodbury Software, 127 White Oak Lane, CN 1001, Old Bridge, NJ 08857, 201-679-0200, $39.95 with refill packs at $9.95. Add $2.50 for first item and $1.00 for each additional item S/H.)

— Robert Jensen

Software **1000/1200/2000/3000**

Probaloto — Not Just Another Random Number Lottery

It's in the news a lot lately: "Local Citizen Wins Millions in State Lottery." With more and more states offering lotteries, *Probaloto* by Gary Olander may be just the ticket for you.

The documentation points out that *Probaloto* is not "just another random number generating lottery program;" the object, however, is to pick the numbers to win you millions. The program comes on a diskette which, while copy-protected, allows backup of the executable file. This backup file must be restored to the original disk for program operation. The only system requirements listed in the documentation are "MS-DOS." The program operated properly with MS-DOS 2.11 on a Tandy 1200HD with 256K RAM. Mr. Olander assured me that the version I received will run on the Tandy 1000, 1200 and 3000 with only 128K, and that a version is also available for the 2000. *Probaloto* is well documented: The program is menu-driven, and so easy to use that after going through the documentation once, you'll probably file it away.

The user is prompted for information about the particular lottery being entered: how many numbers are picked, and from a range of how many numbers. The example given for a regular lottery picks six numbers, from one to 40, but these parameters can be changed. Output can be routed to the screen, or to screen and printer.

The first three selections from the main menu are strictly random number generators. The first picks three numbers (zero to nine), the second picks four numbers (zero to nine), and the third picks six numbers (one to 40), for what is called the regular lottery game.

The fourth selection from the main menu is labeled "weighted" random selections. This is the feature which makes the program more than just a random number generator. The user has the option of setting up "weighted" data files using lists of past winning numbers which are available where lottery tickets are purchased. This is easily done by answering prompts to input the number of times each number (one to 40 or zero to nine) has been a winner. These files can be labeled and saved to disk, and a hard copy can be listed by the printer.

The weighted random selection process can use these data files in one of two ways. If the user feels that since the process used to select winning numbers picked twice as many fours as 17s in the past, it is therefore likely to continue that trend in the future, the program can weight towards the most picked numbers. In the example given, the program would be twice as likely to "randomly" select a four as a 17. If, however, the user feels that Lady Luck plays her hand such that poor old 17 is due, the program can weight toward least picked numbers. The weighted data files operate as if you place numbers in a hat in proportion (direct or inverse) to how many times that number has been picked. Selection from the hat is then random, but weighted. This weighted selection procedure can also be used to set up data files which will favor the selection of your lucky numbers, or conversely, tend to avoid

your unlucky numbers.

To test this weighting process, I wrote a short BASIC program to generate at random regular lottery game winning numbers. I input 60 sets of six numbers into a data file and used this to have *Probaloto* make selections weighted toward the most picked numbers. I then had *Probaloto* make an equal number of selections strictly at random. When these results were compared to my original list, the weighted selections produced 7.5 percent more matches than the random selections. My original list was fairly well distributed. More pronounced variations in frequencies in the data input (such as might occur if the lottery selection procedure is not truly random) would have added to the weighting effect and increased that percentage.

I reviewed the Color Computer version of *Probaloto* for THE RAINBOW several months ago. At that time I indicated that my criticisms of the program were very minor. All but one of these have been corrected in this version (and in the latest Color Computer version). Mr. Olander responded to my criticisms, and I will give PCM readers the benefit of his response to the one which remains.

My criticism is that the data file printout produced by the program is a narrow column near the left margin consisting of numbers and the times each has been picked. For a regular lottery game this column is 40 lines long, using much more paper than would be necessary if the output were better formatted. Mr. Olander points out that not knowing how many numbers are to be printed makes formatting difficult. From my programming experience, I have to agree — but I know it can be done. As Mr. Olander pointed out, the printout for a one to 48 lottery still takes less than one sheet.

If you are into playing a lottery, and believe that the game is truly random in practice as in theory, stick to the random number generator available in BASIC. On the other hand, if you feel that Lady Luck is not always random and want to use some sophisticated mathematics to bring her in on your side, then *Probaloto* may be just the thing for you. I'll second the program's sign-off message: "Good Luck!"

(Gary Olander, 322 Haymarket Pl., Gahanna, OH 43230, 614-475-3315, $29.95)

— Stanley Townsend

Software 1000/1200/3000

Chuckle Pops: Laughs at the Touch of a Key

Every once in a while it happens. Someone comes along with something so different and unique that you just sit back and say, "Wow!"

Lotus Development did it with *1-2-3*. The idea of a spreadsheet *and* graphics was unique for its time. Digital Research's *GEM* software is the same way. Borland International's *Sidekick* rated the same kudos and started a wave of memory-resident programs that is yet to be abated.

OK. So these are some pretty heavy hitters. But there are a lot of others that are "just the thing" to make living with a computer a real joy. And when we see them, we have to tell someone about it. Right away.

Enter *Chuckle Pops*.

I absolutely adore jokes. And jokes are what *Chuckle Pops* is all about. This is a memory-resident program that pops up a joke whenever you need a laugh. There are a couple of hundred jokes in *Chuckle Pops* (and most of them are pretty good) and you can add in your own as well! Rumor has it that there will be new joke disks available, too.

All that is not the point. What is the point is that everyone needs a laugh from time to time. While in the middle of a spreadsheet, a complicated chart design or whatever, all you need to do is press ALT-J (for joke, get it) and a window pops up on the screen to give you a chuckle.

No, *Chuckle Pops* won't set the world on fire, but for $14.95, who can't use some respite from the serious work he or she does with an MS-DOS machine? It works with the 1000, 1200 and 3000. And it works well. Even gives colored windows with a color monitor.

I have had three other memory-resident programs loaded at the same time as *Chuckle Pops* with no conflict.

Try it — you'll like it. And, in the meantime, have you heard the one about . . .

(Enlighten, P.O. Box 2037, Ann Arbor, MI 48106, 800-447-1771, $14.95)

— Lawrence Falk

Software 100

BUSS: Time is Money

Time is money. It may be an old cliche but most everyone would agree that it's true. If you do any type of work that involves charging for time, then the saying is even more valid. If, in addition to charging for time, you also need to be reimbursed for materials used during the job, then *BUSS.BA* comes to the rescue.

BUSS.BA is a billing and time-keeping program. It is useful to anyone who must charge for time and supplies and would like to produce printed records and bills. This includes carpenters, repairmen, mechanics, accountants, consultants and so forth. The program has two separate time clocks that can be punched in and out. Any item can be registered as sold with or without tax. The program can also be used as a simple ledger with expense accounts, income accounts and any number of tracking accounts.

The program is supplied on cassette and comes with a 24-page instruction manual. It requires 6,000 bytes, an additional 1,000 bytes overhead, and eight empty file spaces on the main menu. It will not run on the Model 200. I could not get the program to function properly with Tandy's new portable 3.5-inch disk drive. This was due to interference with the large amount of memory required by the machine language floppy-drive program and *BUSS.BA*. I would expect other machine language programs to cause problems too.

The manual gives complete loading instructions and, once loaded, the program must be customized to the application. The user must edit four program lines to show the name and address of the business, tax rate and the rates for the two time clocks. The step-by-step instructions take the user through this process very easily and I had no problems. As an extensive user

of business programs, I would have preferred this information be kept in a file and not in the program, but the instructions made up for the awkwardness. Once all changes have been made, the new version must be saved to cassette so the changes will not have to be made again.

Once the backup is made and stored, you are ready to run the program. All functions are menu-driven and the main menu is shown in Figure 1.

Figure 1

```
A. NEW ACC.
B. OLD ACC.    J. EXIT                MM/DD/YY
C. DSP. ACC.   K. DEL. ENTRIES
D. PR. ACC.    L. DEL. FILE
E. APP         M. TIME
F. CHARGE TO   N. TIME2                xxxx
G. NOTES                           Bytes Free
H. LABEL       (Copyright 1985 Burkhart)
```

The program is not particularly tolerant of errors and assumes a working knowledge of the Model 100. It is recommended to use all uppercase letters or follow the same format religiously during data entry. This is because the search routine differentiates between upper- and lowercase. I do not like to worry with that kind of detail. During data and note entry, no commas are allowed and there are times when commas are needed. The printer must be online and ready to print when the print option is chosen or the computer locks up.

These things do not make this program unusable, but they are annoying. You cannot edit or delete a single account entry other than by going into the file itself. This can lead to disaster if a typographical error is made or the data format is incorrect. Otherwise, *BUSS.BA* does the job and performs as promised.

For the program to work, there must be at least one account. The input screen is shown in Figure 2. No commas are allowed to be entered into any of these fields. Once all the information has been entered, you are given three options: A. File, B. Make Correction and C. Exit. Pressing A files the entry, pressing B returns to the beginning of entry and allows you to correct a mistake by typing over old information, and C exits to the main menu.

Figure 2

```
NAME
ADDRESS:
CITY:                    STATE:
H:(home phone)      W:(work phone)
APP: (appointment or comment)
JOB LOC.:
```

The OLD ACC option allows the user to search the database and retrieve a previously entered account. This is also the option to activate an account. You must activate a particular account for printing, deleting, charging to and so on. This option prompts with FIND? and you type in any portion of any field in the database. The program displays all accounts that match (one at a time) and the user chooses which one to use. That account becomes the active customer account.

The DSP ACC option allows viewing billing entries for the active customer account. The display shows each entry with item total then summarizes with total tax, grand total and balance at the end of the display. If there are more

entries than will fit on the screen, the program pauses until a key is pressed to allow viewing.

The PR ACC option produces a paper record of the account activity with total tax, total and balance. Be sure to have the printer ready or the computer locks up.

When the E APP option is selected, the computer searches the files for entries made in the APP: field. If there is an entry, the computer displays it. I am not sure what good this feature is, but maybe some enterprising user will come up with something. The manual gives no hints and I found no use for it in my business.

CHARGE TO is where the real day-to-day action takes place. First activate a file to charge to, then the program prompts with a 1. This is a mark-up multiplier and if the user presses ENTER then 1 is used. Enter 1.5 to mark up something 50 percent or 0.8 to mark down 20 percent and so on. The next item to be entered is whether the item is taxable or not. The user is then asked to enter a description (up to 29 characters long) of the item. The next prompt is QUANTITY and a suitable number should be entered. Lastly, the unit price is entered, the program computes and displays the entry and asks if it is correct. If the user answers Y, the entry is filed. The program then loops back to the sales tax prompt. If you want to change the multiplier, exit to the main menu and reenter the "charge to" option.

The NOTES option allows the user to attach notes to a particular account. An account must be activated before accessing this option. When it is accessed, the menu appears as in Figure 3. It allows the addition, deletion and printing of notes. The notes can be edited by going directly into the NOTE.DO file. The note feature can be used to store important information about a client or account that will not fit on the main record.

One problem I found is when the program is first loaded and initialed, any current NOTE.DO file is destroyed. The manual does not warn of this danger. If you have important names and phone numbers in the NOTE.DO file, rename or save it. The manual also cautions you not to exceed 230 characters per note or risk losing all data the next time notes are deleted. I feel a danger such as this should be eliminated by the program itself. Data loss is serious and I do not have time to count the characters in my notes. I do appreciate the warning, but this area could be improved.

Figure 3

```
A. DEL. (filename) ENTRIES
B. WRITE TO (filename)
C. READ FROM (filename)
D. PR. (filename)
E. EXIT
```

The LABEL option is used to print mailing labels for the active account or return address labels. The user is prompted NUMBER OF LABELS? and, after entering a number, the user chooses return address or customer labels or exit. When printing customer labels, a comment line may be added if desired. This comment is not available for return address labels. It would be nice for seasonal advertising slogans and the like. This option does not feature any way to align the labels before printing so you have to test it on live data to see if it prints in the proper location. I would have liked to print labels for all customers, not just the active customer account. Maybe this feature will be added in future releases.

DEL ENTRIES functions the same as the delete note function except that it deletes all entries for the active account.

DEL FILE is similar to the other delete options except that it deletes all notes, account entries and the file itself.

The TIME and TIME2 options are similar to each other, but can charge different rates. These rates are determined when the program is first initialized. To clock in on a job, first select and activate the appropriate file and then choose the appropriate time clock. The time clock runs when you are in the program and even if the computer is turned off. The manual says that it does not work properly if used through midnight.

The time clocks are one of the best features of the program, but I would like to see several improvements. I would like to know which clock is charging what rate by looking at the menu and for the rate to print on the bill. As it is, the time charged for is displayed in hundredths of an hour along with the date, a comment and the total charge. Also, I would like to know if the time clock is "punched in" or not. While using it, I forgot several times to "punch out," but if there had been a reminder, I might not have forgotten. No program can overcome operator stupidity, but helps and reminders are always welcome.

The program has a lot going for it. It is extremely useful to me in my consulting business and I can see the program paying for itself many times over with correct billing of time spent and material used on the job. The program is also useful for tracking travel expenses and other accounts that are easily overlooked during the hectic work day. *BUSS.BA*, in combination with a portable printer, allows the user to prepare and present bills for work on the spot, which makes for more immediate payment and better cash flow.

If you have a job that involves billing for time and/or material, this program deserves your attention. It can improve work efficiency and save time, which can add up to big money.

(Ronald F. Burkart, Rt. 3 Box 883, Hillsborough, NC 27278, $89.95)

— Tim Birtcher

Software **1000/1200/3000**

The Factory: Problem Solving Made Fun and Challenging

The very mention of problem-solving makes most students groan, so don't tell them this learning tool has anything to do with problem-solving. *The Factory* is too much fun to be pre-judged and ruined with negative connotations.

Making learning fun is a challenge for teachers; and dealing with the needs of a student who can finish an assignment quicker than a wink can be an even greater one. *The Factory* eases both challenges. If computers and software like *The Factory* from Sunburst Communications had been available when I

faced these challenges as a teacher 20 years ago, my job would have been much easier and more fun.

In the hands of a creative teacher using the Teachers' Guide, which comes with the program, *The Factory* has much to offer in the classroom. I was impressed with how much can be learned with no instruction whatsoever.

reproducing designs created by the computer, he chooses from three levels of difficulty. The easiest level is simple enough for a first or second grader; the most difficult, fun for an adult.

If the product is not a perfect match for the one he is duplicting, he is told that there is a flaw in the product and asked if he would like to try again. If he chooses to try again, the machines are cleared and he begins from scratch. This was the only area of the program I would have changed. In my opinion, the program would be improved if a choice was given here to either begin over or to change individual machines. Time and again on the more complicated designs that involved several steps, it would become quickly obvious that changing just one of the machines would give the desired results. But, unfortunately, the only choice is to start over and re-create the machines that were already correct.

"The Factory *has much to offer in the classroom. I was impressed with how much can be learned with no instruction whatsoever.*"

Experimenting with choices from the main menu leads a player through a series of steps that reveals enough about the game to preclude how-to instructions. First, the student learns how to design machines and organize them in an assembly line. Then he sets them in motion and waits to see the outcome of his product.

The challenge comes when he tries to reproduce a product the computer or another player has designed. If he is

After finishing with *The Factory*, the player has the option of turning off the computer or going to another diskette. If another diskette is chosen, the computer boots up after the new diskette is inserted. This is a handy feature for young students who are unsure how to get a new program started or how to turn the computer on or off. For example, the program instructs: "To turn the computer off: 1) Remove the diskette from the drive and put it in a safe place. 2) Turn the computer off. The switch is located on the right side of the computer. 3) Turn off your monitor." The directions are just as clear and direct for removing *The Factory* and getting another program started.

The Factory deservedly received recognition from The Council for Exceptional Children and the Learning Periodicals Group. Designed for grade levels four through nine, (using a second grade reading level) and created with an interest level for any age group, *The Factory* reaches its objectives in helping develop visual discrimination, spatial

perception and logical thinking. Be forewarned however, it does require 256K.

I decided to test the program outside the age range for which it was designed. I used a gifted second grader and two high school students to see if *The Factory* has a level of interest that appeals to all age groups. The second grader had great fun designing products for older kids to reproduce, and the high school students enjoyed the challenge. (I made the mistake of putting *The Factory* on the computer at the office, and I lost my secretary for the day.)

The Factory is a fun teaching tool for any classroom. Its qualities make it especially beneficial in working with exceptional students. It presents a challenge to the gifted student by way of a game that can be used independently for moments of creative learning, and at the same time, it has equally as much to offer the slow learner. Choosing the level of difficulty is the key.

The Teachers' Guide is short, sweet and to the point. It provides ideas and worksheets to help a teacher get the most from the program. I recommend *The Factory* for the classroom. But don't take it to the office if you want your secretary to get any work done.

(Sunburst Communications, Inc., 39 Washington Ave., Pleasantville, NY 10570; 800-431-1934, $59, lab of 10 disks $177)

— Irma S. Canfield

Hardware 100/200/600

Shut out Eyestrain with Portable Light

Surely I'm not the only Model 100 user who checks to see if there's an adequate supply of aspirin, along with dependable batteries, packed in the lug-everywhere bag. Between contact lenses, Max Factor and little PoCo's obstreperous LCD display screen, my eyes often enjoy taking their revenge on the old braincase. Enter Portable Light from AMRO Computer Services to ease the situation.

Portable Light is a small, compact, battery-operated (two AA's) light source weighing approximately three ounces. Portable Light has a six-inch flexible neck to provide any height or angle variation necessary to light up and make legible a portable computer screen. There is a detachable base/suction cup to mount the light on any horizontal surface, such as the briefcase in your lap (under the Model 100) while your lap is being transported somewhere by car, plane, etc. Portable Light also has a clip for fastening it to surfaces other than horizontals such as clipboards or car visors. Its compact and versatile structure, ability to ease eyestrain and inexpensive price tag give me ideas (OK, daydreams) about using it on my camping excursions as well as the annual tango with my car's points and plugs (oh yes, the contacts and Max Factor will be there with me too).

(AMRO Computer Services, Dept. F, P.O. Box 1131, Tualatin, OR 97062, 503-692-5926, $6.99)

— Monica Dorth

Software 100

Tape is Tape, but Now it's *FAST*

Cassette tape is not the ideal choice for data storage, but it is cheap, readily available and portable. It is also extremely slow. Slowest of all is the process of transferring documents to and from tape. To remedy the situation, various solutions, like the Radio Shack Disk-Video interface, have appeared on the market. Along with other drawbacks, most of these devices are expensive. So in zips *FAST* to salvage your cassette recorder, your savings account and your midnight oil.

FAST doesn't claim to do much; just speed the process of SAVEing and LOADing .DO files to tape. It performs exactly as promised by operating roughly three times as fast as the 100's ROM routines. That may not seem like much unless you save and load numerous text files. Then it's over a minute for each of my 5K files saved.

PCM

☐ Please send me the following back issues:

(Payment must accompany back issue orders.)

NO.	MO.	YR.		PRICE	
			VOLUME 1		
1	JULY	'83	PREMIER ISSUE	$3.00	☐
2	AUG.	'83		$3.00	☐
3	SEPT.	'83		$3.00	☐
4	OCT.	'83		$3.00	☐
5	NOV.	'83		$3.00	☐
6	DEC.	'83		$3.00	☐
7	JAN.	'84		$3.00	☐
8	FEB.	'84		$3.00	☐
9	MAR.	'84		$3.00	☐
10	APR.	'84	LIMITED	$3.00	☐
12	JUNE	'84		$3.00	☐
			VOLUME 2		
2	AUG.	'84		$3.00	☐
3	SEPT.	'84		$3.00	☐
6	DEC.	'84		$3.00	☐
7	JAN.	'85		$3.00	☐
8	FEB.	'85		$3.00	☐
9	MAR.	'85		$3.00	☐
10	APR.	'85		$3.00	☐
11	MAY	'85		$3.00	☐
12	JUNE	'85		$3.00	☐
			VOLUME 3		
1	JULY	'85		$3.00	☐
2	AUG.	'85		$3.00	☐
4	OCT.	'85		$3.00	☐
5	NOV.	'85		$3.00	☐
6	DEC.	'85		$3.00	☐
7	JAN.	'86		$3.00	☐
8	FEB.	'86		$3.00	☐
9	MAR.	'86		$3.00	☐
10	APR.	'86		$3.00	☐
11	MAY	'86		$3.00	☐

SHIPPING & HANDLING _____

KY RESIDENTS ADD 5% _____

TOTAL AMOUNT ENCLOSED _____

NAME _____ ADDRESS _____ CITY _____ STATE _____ ZIP _____ ☐PAYMENT ENCLOSED

☐ ☐ ☐ ☐

CARD # _____ EXPIRATION DATE _____ PHONE # _____

SIGNATURE _____

Installation is a matter of loading two BASIC programs from tape. SAVER and LOADER now take the place of the TEXT program's equivalent functions. The brief documentation is more than adequate for installation and operation.

Using *FAST* is almost as easy as installing it. Other than a few minor quirks, *FAST* is practically self-explanatory. *FAST* gets its speed by saving and loading documents as if they were BASIC files. Instead of the constant start/stop of normal cassette saves, *FAST* saves and loads without any pauses. The only drawback is that document files currently on tape cannot be retrieved with *FAST*. They must be stored on tape with the SAVER program first.

The extra speed provided by *FAST* does not mean that you must sacrifice reliability. Each time SAVER writes a document to tape, it also calculates and stores a checksum. When the document is loaded from tape, this checksum is again computed and compared to the value on tape. Any variation is reported by *FAST* to alert you to possible problems.

SAVER automatically stores the last text file you were working on. Simply select the program from the menu, and SAVER will display the first line of the text file and the length in bytes. You can choose to assign a name, or press ENTER to have SAVER use the first six characters in the text as the filename.

SAVER can also back up all .DO files on the machine at once. When the program prompts for a filename, pressing the period (.) key tells SAVER to backup all text files in a single file on tape. SAVER assigns a name to the file such as "B01/31" where the 'B' indicates backup and the "01/31" is the month and day (e.g., January 31) of the backup.

Loading *FAST* saves is just as easy. Choose the LOADER program from the menu and follow the instructions. You can load a file by name or press ENTER at the prompt to search the tape. LOADER will then pause at each file, display its length and give you the option to load it or continue the search. Once you instruct LOADER to load a file, it asks for a name. You can supply a name or press ENTER and LOADER will save it as TAPE.DO. If you specify a file that already exists, LOADER will add the new file at the end.

One very useful capability that *FAST* provides is a means of recovering text files after a cold start. If you've ever lost 20K worth of text as a result of an accidental cold start, then you know why they call it "cold." Actually, as the *FAST* documentation explains, the text isn't gone, it's just inaccessible. If you remember not to go to TEXT and begin writing, then you can save it all. You save it by dumping the entire contents of memory to tape. Depending on the amount of memory in your machine, this will either take two or three successive CSAVEM commands, as described in the *FAST* manual.

Using LOADER, you restore each chunk to a text file. Then you begin the process of weeding out garbage characters and separating the text files. This isn't as bad as it sounds; *FAST* puts the black box symbol (GRPH-SHIFT-X) between each file.

By using TEXT's editing facilities, you can find each black box symbol, select the text and CUT it to the paste buffer. If the text is a document that you want, you can open a new text file and PASTE it there. Otherwise, you continue the process with the next section. This same procedure works with *FAST* backup files where all .DO files are stored together in a single backup. Each is separated by the graphic black box.

FAST does have limits. It only works with text files, so unless you save a lot of text, *FAST* isn't for you. SAVER only saves the last text file worked on, so to save any other file, you must first "visit" it as though you were going to edit it and then run SAVER. SAVER uses the beginning characters in the file and not the filename as the default document name on tape. Changing the program to use the current name would be a welcome improvement. LOADER offers a very nice option to search the tape; unfortunately, it only loads one file for each search. Once you load a file, LOADER returns to the Model 100's menu. It would be a real plus to load several files without restarting the program.

FAST delivers as promised; nothing more and nothing less. *FAST* won't take the place of a disk drive, but it will greatly improve cassette storage at a fraction of the cost. So if you're not ready to abandon tape or can't afford to, now you can have your tape and beat it too.

(Minsof, Inc., P.O. Box 1153, Minneapolis, MN 55440, supplied on cassette $19.95)

— Dennis Kirley

Forecasting with *Fourcast*

Fourcast is a business forecasting program for MS-DOS microcomputers. Based upon the principles of time series analysis and using a method known as a Moving Window-Spectral (MWS), the program is designed to extract patterns or cycles from input data and to predict, via statistical tables and graphs, the most probable future trend of the data.

The program is controlled through four menus: Main Menu, File Operations Menu, Forecasting Menu and Report Menu. The main menu lists 15 options:

H	Help explanation of data input modes
D	Specify run description code
V	Specify list of variables in this run
C	Enter the create mode
U	Enter the update mode
L	List input data
P	Plot input data
O	Perform file operations
X	Data conversion to fourcast files
N	Network intercomputer communications
S	Calculate system parameters
B	Plot system parameters
F	Forecast with current system parameters
R	Print final report
T	Terminate this session

The first option, H, evokes an on-screen synopsis of input modes, variables and data structure of the program. These are called by the next four options of the main menu: D, V, C and U.

The D option allows you to give a name to the analytical run. The name is used as a title for printed reports and has no effect on the data or the computations.

The V option is used to specify which data files are to be used in the session. If you don't enter a name, the program defaults to the last file run. Of course, a new set of data requires a new list of variables. As an example, *Fourcast* comes with a sample case study (the description code), which contains three data files: SALES---.emc, PRICE---.emc and INCOME--.emc. These files contain all the information necessary to examine and test the features and functions of the program. If only one set of data is to be entered, then only one variable is identified (univariant). In the example three files were created because there were three different sets of information (multivariant).

The C option is self-explanatory. It is used to create a new data file or files for analysis. When chosen, the program goes through the sequence to create a new data file. You first enter a filename, under which the data will be stored. Second, a description is requested. The description has no effect on the calculations. Next, the program asks for the total number of data points to be entered. The maximum is 100, which is a rather small amount for an in-depth statistical study. The last entry is the length, in number of data points, of the moving window, or the portion of the data that is to be used for the computations and predictions. The manual suggests, "Best results are most likely when the length of the window corresponds to the number of time periods in the dominant cyclical component of the time series." This suggestion assumes that the user has some knowledge of the expected cycle of the information. And here is where a problem arises.

The first sentence of the manual states, "*Fourcast* is very easy to use and excellent results can be obtained by anyone." From the standpoint of physically operating the program it is easy to use, but in order to choose the length of the moving window, the user must have some knowledge of numerical analysis. The program's algorithm is designed based on numerical analysis. If you do not have this knowledge, you will not know what "dominant cycle" is appropriate for the data being entered. The length of the moving window is critical in obtaining the most statistically significant results and a mere guess is inappropriate. Therefore, a contradiction seems to exist between what is required of the user and what is claimed in the manual's introduction. Although the manual is not intended to be a mathematical primer, the importance of this topic is given but a few brief paragraphs.

Option U of the main menu is for updating an existing data file. Since the data is always entered with the date, it is easy to find the datum for edition or deletion. New data may also be inserted anywhere within the file. The program automatically puts the new data in its proper position.

The L and P options list and plot the input data and give you the opportunity to view a table or graph of the entered information and to check its accuracy.

The second menu of *Fourcast* is called when the O option (File Operations) is selected. The following selections are displayed:

M	Multiply the contents of two files
S	Subtract the contents of one file from the contents of another
D	Divide the contents of one file by the contents of another
R	Take reciprocals of the contents of a file
L	Take natural logarithms (base e) of the contents of a file
L10	Take logarithms (base 10) of the contents of a file
IL	Take inverse natural logarithms of the contents of a file
IL10	Take inverse logarithms of the contents of a file
XC	Multiply the contents of a file by a constant
AC	Add a constant to the contents of a file
Q	Quit file operations & transformation mode

These operations are used with the graphs printed from the R option of the main menu which is Print Final Report. The file operation functions are used as a means of rescaling the input data so that the forecast data can all be plotted within the limits of the Y-axis screen dimension. Any file operations, however, must be made on the input data before the forecasting computations are begun.

Following the O option of the main menu is the X option, (Data Conversion to *Fourcast* files). The manual makes no mention of this option. When this option was selected the following message was displayed on the screen:

```
DATA CONVERSION TO FOURCAST FILE
Data conversion routine must be requested from EMC
```

Press ENTER to continue, followed the message, and when pressed the main menu reappeared on the screen. Therefore, the function of this option is unknown.

The program does not limit data entry to the keyboard. Through the N option, (Network Intercomputer Communications) the program accepts data from another computer through a modem. This is a handy feature, particularly if you want to download information from a mainframe computer database. The downloading procedure is simple. The program establishes the communications link and, through the use of the CTRL S key combination as an on/off toggle switch, captures the incoming data in a file named *DOWNLOAD*. In order for this file to be used by the program, it is necessary to rename it through MS-DOS (the RENAME command) and then, using a word processor, edit it until it matches *Fourcast*'s data format. This conversion procedure is rather involved compared to the simplicity of the downloading operation.

After all the data is entered, the S option is used to begin the program's computations. After the prompt is answered, the program loads the data and displays it in a table, allowing you to view the information. After another continuation prompt, the program performs a time/frequency transformation and a parameter estimation mode. It then displays, in a tabular format, the statistical analysis of variance. It includes the following measures for each frequency computed: the total sum of squares (SST), the sum of squared errors (SSE), the Rsquared statistic (Rsqrd), the mean square for the model (MSM), the mean square error (MSE), and the F value, which is calculated from the MSM/MSE ratio and indicates the "goodness" of the statistical fit of the data to the computed model.

Following the variance table is the parameter estimates table. This table displays four more statistical measures relating to the frequency of the data: the magnitude value (either positive or negative) is the amplitude change of the cyclic components of the data (as based on sinusoidal waves); the shift value is a measure of the phase shift of the cycle, or its starting point in the data; the S number measures the standard error of estimate in the companion parameters; and t, the last value, tests the significance of the parameter being estimated.

Next, the program analyzes residuals

computation. A residual is another statistical tool that measures how the fitted data differ from the actual data. After the computation, a table is displayed showing the date, actual datum, fitted datum, the residual and the percent error of the residual to the actual data. The critical information is the percent error values. If small, then the fitted data is in good agreement with the actual data. Otherwise, the fit is not so good. Nevertheless, a good fit does not guarantee the validity of the forecast.

After the residual table, the residuals are plotted on the screen in a simple graph. The values of the residuals are plotted against time with asterisks denoting the plot positions. According to the manual, there should be no discernible pattern in the asterisks if the analysis is valid. If a pattern emerges, then a relevant variable in the data is absent and the data must be re-examined, on a trial and error basis, using some other procedure than is available from *Fourcast*.

The last function of the program shows a summary statistical table. Along with the SST, SSE, Rsqrd, MSM, MSE and F measures, a DW statistic is printed. This is the Durbin Watson statistic and it is used to measure the durability of the unsystematic residual variation assumption under a test of significance. The manual gives a brief explanation of how to use the DW value.

With all of its functions, the S option obviously represents the computational heart of the program. Just as obvious is that the interpretation of this statistical information is not a simple feat. If you are not an expert, then much of the information will have little meaning for you. Certainly, the claims that anyone can use the program is stretching the truth, for without a working knowledge of statistics, one would be left on a mental precipice.

The next option of the main menu, B, plots the system frequency parameters and the impulse response. The frequency graphs that are plotted are described by the manual as ". . .useful for observing the way in which the parameters change over different frequencies . . ." The graphs are not clearly explained in the manual and the information they present is open to interpretation. As the manual states, "Note that time series in the sample case study are very short in length. Not very much significance can be placed on the relationships portrayed."

Following the frequency graphs are the impulse response plots. An impulse is an event of short duration that may have a long term impact on a trend. An analogy would be the 1974 Arab oil embargo and its long term effect on gasoline prices. The impulse response graphs are, therefore, only theoretical representations. Here again, these graphs are not easily interpreted.

The most important option of the main menu, indeed the main objective of the program, is F option (Forecast with Current System Parameters). The third menu has the following options:

F	Automatic forecast
I	Interactive forecasting (with current overrides if any)
O	Specify overrides in time periods t (t=1,. . . . . , 20)
T	Tabulate current trial forecast
P	Plot current trial forecast
C	Change number of forecast periods
Q	Save current forecast and quit the forecasting mode

The Automatic forecast option, F, computes the file data for forecasting. After the calculations are completed, the program returns to the forecasting menu.

Options I and O work in conjunction with each other. In order for the interactive forecasting, option I, to be used, it is necessary to first specify the override in time periods using option O. Then the interactive forecasting function recalculates the data and returns to the menu.

After selecting option T (Tabulate current trial forecast) a table is displayed on the screen listing the datum number, datum date and the calculated forecast value. The table should indicate any major trends in the data, but a graph is easier to understand and that is the purpose of the next menu selection.

Option P plots the forecast data against future time on a graph. In the sample case study, sales, price and income data are each plotted twenty forecast periods into the future (twenty years in this particular case). This graph is the main objective of *Fourcast* and any trends are easily seen in this representation.

By choosing option C, you can alter the number of forecast periods. The allowable range is from one to 65. If the periods are changed, the forecast data is recomputed.

When you're finished with the forecasting menu, the Q option saves all the data previously calculated and returns to the main menu.

The next to last option of the main menu is R (Print Final Report). This option calls the fourth and last control menu of the program:

A	Forecast assessment
P	Plot forecast
T	Tabulate forecast
Q	Quit the report mode

When option A (Forecast assessment) is selected, the program prints a table containing the datum number, the datum date, the actual datum value, the forecast value and two statistical measures: %ERROR and MAPE. The %ERROR value is simply the percentage difference between the forecast value and the actual value, while the MAPE (Mean Absolute Percentage Error) is the average of the cumulative forecast error for a forecast time period.

After the table is displayed, the program displays a graph on which both the actual and forecast values are plotted for each forecast period. The graph gives a good idea of the relationship between the two.

The P option allows viewing the forecast data via an extended graph. If your computer is equipped for high resolution graphics, the program will plot the graph in that mode. Otherwise, a full-screen ASCII graph is shown. The graph, or graphs in the case of multivariables, provides significant information about the future behavior of the data.

When the T option is selected, the program prints a forecast table showing the upi (upper limit of 95 percent prediction interval) value, the computed forecast value and the lpi (lower limit of 95 percent prediction interval) value for each forecast time period, including the forecast date. This table provides critical information about the statistical

validity of the computed forecast values.

The last option, Q, takes you back to the main menu. At the main menu, the T option exits the program and returns you to MS-DOS.

Fourcast's main and most powerful feature is the Forecasting Mode, as summoned by choosing the F option from the main menu. Its operation and resulting analyses were obtained through the sample case study provided on the program's diskette. However, there is one situation which must be mentioned: The forecasting mode does not work with any data other than the sample case study without first connecting to a special communications host system maintained by Engineering Management Consultants, the program's developers. This communications system, called the Electronic Fourcast Service, is accessed via a modem. What the service does is unlock the Forecasting Mode of the program so you can run your own data. Once the forecasting menu is displayed, you can disconnect from the service. Each time you run the program and wish to use the forecasting functions, you must reconnect to the service to unlock the menu.

In order to use this service, you must fill out the application form in the manual and mail it in to obtain an identification code and password. Without them you cannot access the service and, therefore cannot use the Forecasting Mode of *Fourcast*. You must pay for the access time, although the first hour is free.

This unlocking procedure is a unique approach to program security, if it is indeed that. Regardless of why this particular technique was chosen, it can run into unforeseen costs. If you don't have a modem or an RS-232 board, then the program is of limited use and communications equipment must be purchased. On the other hand, you may have the necessary hardware but do not wish to pay a monthly connect fee. The initial cost of the program may be minuscule in comparison to its operating costs — a serious point to consider.

Another point, perhaps more important, is the continued availability of the electronic service. Without the service, the program's forecasting function cannot be accessed and is valueless. The unlocking technique is a drastic step for the sake of program protection, and is, in my opinion, a serious drawback.

Although the manual is a brief 69 pages in length, it covers the important aspects of the program's operation. Each menu is presented and discussed, including tables and graphs of the sample case study data. The mathematical relationships of the various statistical measures are defined and explained, although the manual is not intended to be a text on statistical analysis. It does provide you with a reasonable basis for further study. Five reference books are listed for that purpose. The best approach to understanding *Fourcast* is to work through the manual while running the program.

If you are in the market for a sophisticated business forecasting program, then you should consider *Fourcast*. Remember, it is a tool that requires a basic knowledge of statistical analysis in order to take full advantage of its potential.

(Engineering Management Consultants, P.O. Box 312, Fairfax, VA 22030, $30)

— Ralph Rideout

New Products

The following products recently have been received by PCM, examined by our magazine staff and approved for the PCM *Seal of Certification*, your assurance that we have seen the product and have ascertained that is what it purports to be. This month the *Seal of Certification* has been issued to:

BDL.Estate, helps you organize your estate — what you own, what you owe, where you work, military service, where everything is and who's important. Requires Tandy 1000, 1200 or 3000. *BLD Homeware, 2509 North Campbell Avenue, #328, Tucson, AZ 85719, (602) 557-1435, $69.95.*

DoubleDOS, allows you to concurrently run two applications in memory and switch between the running programs. Requires Tandy 1000, 1200 or 3000. *SoftLogic Solutions, 530 Chestnut Street Manchester, NH 03101, $49.95 plus $5 S/H*

FAST, greatly decreases the time needed to save and load text files to and from tape. Requires Model 100 and cassette recorder. *Minsof, Inc., P.O. Box 1153, Minneapolis, MN 55440, cassette $19.95.*

Media Master, a disk-to-disk format conversion program that provides file compatibility between over 70 different CP/M and MS-DOS computers. Requires Tandy 1000, 1200 or 3000. *Intersecting Concepts, 4573 Heatherglen Court, Moorpark, CA, (805)529-5073, $39.95.*

Mini Magic Card, memory card that allows you to expand memory up to 576K with one, short card. Requires Tandy 1200 or 3000. *Everex Systems, Inc., 47777 Warm Springs Boulevard, Fremont, CA 94539.*

PC-Key Draw, a graphics design program for CAD and graphics design applications. Requires Tandy 1000, 1200 or 3000 with graphics option. *OEDWARE, P.O. Box 595, Columbia, MD 21045-0595, $100.*

Resident, allows you to make almost any MS-DOS application program into a memory-resident program that may be called up at any time with a single keystroke while other applications are running. Requires Tandy 1000, 1200 or 3000. *Information Software, Inc., 2639 Walnut Hill Lane #135, Dallas, TX 95229, (214) 353-2966, $89.95 plus $5 S/H.*

What'sBest!, automatically generates best-possible solutions to spreadsheet models. Requires Tandy 1000, 1200 or 3000 and Lotus Symphony or 1-2-3. 256K required for Commercial Version, 640K for Professional Version. *General Optimization, Inc., 2251 North Geneva Terrace Chicago, IL 60614, (312) 248-7300, Commercial: $695, Professional: $995.*

Video Title Editor, allows add title and credit screens and various displays to home-recorded videotapes. Requires Tandy 1000, 1200 or 3000, color graphics option and VCR. *Videoware, 1977 W. Twelve Mile Road, Suite 180, Southfield, MI 48076, (313) 626-7208, $29.95.*

Zuckerboard Expansion Memory Board, adds 512K to a Tandy 1000 on a single board, bringing total system memory to the maximum 640K. Socket for optional clock/calendar. Requires Tandy 1000. *Advanced Transducer Devices, Inc., 1287 Lawrence Station Road, Sunnyvale, CA 94089, (408) 734-4631, $149.*

By awarding a *Seal*, the magazine certifies the program *does exist*, but this *does not* constitute any guarantee of satisfaction. As soon as possible, these hardware or software items will be forwarded to PCM's reviewers for evaluation.

Using *BAREAD 2.1*

Bar code listings must be read in numerical order beginning with Line 1 and continuing through the last line of the listing. The computer display is used to prompt you as to which line to scan and give you warning messages should you happen to get out of step.

When you run *BAREAD*, it asks you to scan the first line of the bar code listing. This line contains the name of the program as well as the beginning of the program itself. The computer will sound a high-pitched beep whenever it's ready for you to scan a line. After a line has been successfully read, you'll hear a lower beep. A "blip-bloop" sound prompts you to turn your attention to the screen for a message. You'll hear this when you accidentally scan a line out of sequence.

After reading the first line, you continue scanning with the second line. Remember to wait for a high beep before scanning and then listen for a low beep to indicate a successful read.

Once the last line of the listing has been scanned, *BAREAD* will return control to the Tandy 100/200 menu screen. Note that the program you just scanned is now in the directory with a .DO extension.

The final step is to convert the .DO text file to a normal BASIC program. This is done quite simply by going to BASIC and loading the file with a command such as LOAD"TEST .DO" (if the program name were TEST). The program will load into BASIC and will be ready to run. To save the program in BASIC's compressed format (.BA extension), you'd type SAVE"TEST" (if the program were named TEST). You may then kill the .DO file with KILL "TEST.DO".

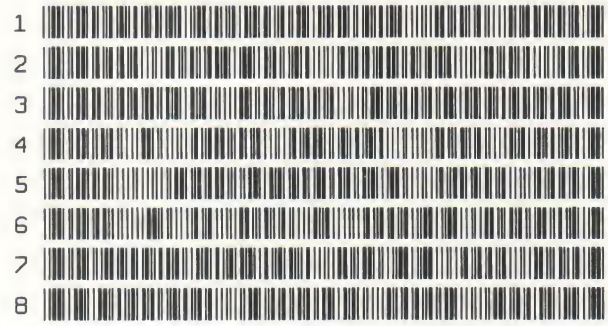
BAREAD 2.1

```
1000 ' *** Initialize ***
1010 ON ERROR GOTO 1040
1020 CLEAR 1000:MAXFILES=2
1030 GOTO 1050
1040 IF ERR=5 THEN RESUME NEXT
1050 ON ERROR GOTO 0
1060 RUNM "B30F9"
1070 OPEN "WAND:" FOR INPUT AS #1
1080 UC%=-1
1090 PC$="0123456789ABCDEFGHIJKLMNOPQRST
UVWXYZabcdefghijklmnopqrstuvwxyz- $+"
1100 DIM RW$(36)
1110 ER$(1)="You must scan line 1 first!
"
1120 ER$(2)="You've SKIPPED a line!"
1130 ER$(3)="You've ALREADY SCANNED this
 line!"
1140 ER$(4)="Code not PCM2/39 format!"
1150 ER$(5)="Command not applicable here
!"
1160 ER$(6)="You cannot skip this line!"
1170 ER$(7)="Selected resume file not in
 computer!"
1180 ' *** Read Reserved Words List ***
1190 DATA BEEP,CLEAR,CLOSE,DATA,DEFDBL,D
EFINT,DEFNG,DEFSTR,ELSE,GOSUB,GOTO
1200 DATA INKEY$,INPUT,INSTR(,LCOPY,LEFT
$(,LINE(,LOADM,LPRINT,USING,MAXFILES
1210 DATA MID$(,NEXT,PEEK,POKE,POWER,PRE
SET(,PRINT,READ,RESTORE,RETURN,RIGHT$(
1220 DATA SOUND,SPACE$(,STRING$(,THEN
1230 FOR I%=1 TO 36:READ RW$(I%):NEXT I%
1240 ' *** Procedure Begins Here ***
1250 CLS:PRINT@44,"PCM Bar Code Program
Reader v2.1"
1260 LINE(20,4)-(219,18),1,B:LINE(22,6)-
(217,16),1,B
```

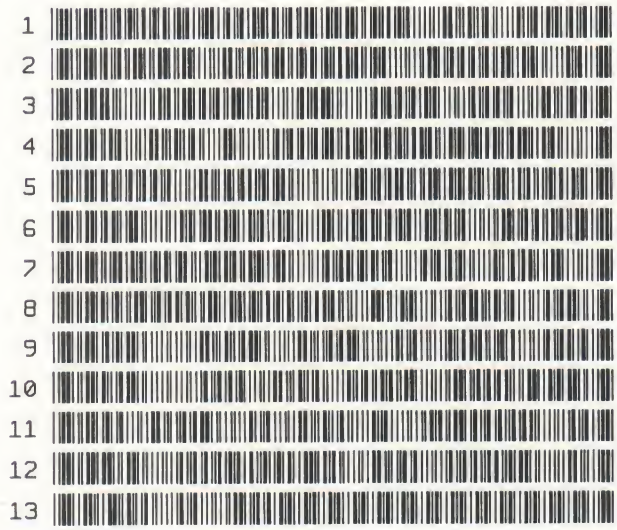
```
127Ø NN%=1
128Ø GOSUB 166Ø:IF ER%>Ø THEN GOSUB 162Ø
:GOTO 128Ø
129Ø IF LL%=Ø AND INSTR("YN",IL$)>Ø THEN
 ER%=5:GOSUB 162Ø:GOTO 128Ø
13ØØ IF LL%=Ø THEN ON INSTR("ALSR",IL$)
GOTO 182Ø,189Ø,198Ø,2Ø5Ø
131Ø IF LL%=1295 THEN 135Ø
132Ø IF LL%<>NN% AND NN%=1 THEN ER%=1:GO
SUB 162Ø:GOTO 128Ø
133Ø IF LL%<NN% THEN ER%=3:GOSUB 162Ø:GO
TO 128Ø
134Ø IF LL%>NN% AND NN%>1 THEN ER%=2:GOS
UB 162Ø:GOTO 128Ø
135Ø IL$=RIGHT$(IL$,19)
136Ø IF LL%=1 AND NN%>Ø THEN GOSUB 178Ø
137Ø CL$=CL$+IL$
138Ø FOR I%=1 TO LEN(CL$)
139Ø     CH$=MID$(CL$,I%,1)
14ØØ     IF CH$="%" THEN GOSUB 151Ø:IF NL
% THEN 147Ø ELSE GOTO 144Ø
141Ø     IF CH$="/" THEN GOSUB 155Ø:IF NL
% THEN 147Ø ELSE GOTO 144Ø
142Ø     IF CH$="." THEN UC%=NOT(UC%):GOT
O 145Ø
143Ø     IF CH$=>"A" AND CH$<="Z" AND NOT
(UC%) THEN CH$=CHR$(ASC(CH$)+32)
144Ø     XX$=XX$+CH$:IF RIGHT$(XX$,1)=CHR
$(13) THEN PRINT#2,XX$;:XX$="":UC%=-1
145Ø NEXT I%
146Ø CL$=""
147Ø PRINT@2ØØ,SPACE$(8Ø);
148Ø IF LL%<>1295 THEN NN%=LL%+1:GOTO 12
8Ø
149Ø ' *** Done ***
15ØØ CLOSE:CALL 618Ø7!:CLEAR 5ØØ,HIMEM:M
ENU
151Ø ' *** Decode Reserved Word ***
152Ø NL%=Ø:IF I%>LEN(CL$)-1 THEN NL%=-1:
CL$="%":GOTO 154Ø
153Ø I%=I%+1:CH$=RW$(INSTR(PC$,MID$(CL$,
I%,1)))
154Ø RETURN
155Ø ' *** Decode Hex and Control Charac
ters ***
156Ø NL%=Ø:IF I%>LEN(CL$)-1 THEN NL%=-1:
CL$="/":GOTO 161Ø
157Ø I%=I%+1:IF INSTR("/%.",MID$(CL$,I%,
1))>Ø THEN CH$=MID$(CL$,I%,1):GOTO 161Ø
158Ø IF I%>LEN(CL$)-1 THEN NL%=-1:CL$=RI
GHT$(CL$,2):GOTO 161Ø
159Ø HX$=MID$(CL$,I%,2):CH$=CHR$((INSTR(
"Ø123456789ABCDEF",LEFT$(HX$,1))-1)*16+I
NSTR("Ø123456789ABCDEF",RIGHT$(HX$,1))-1
)
16ØØ I%=I%+1
161Ø RETURN
162Ø ' *** Error Codes ***
163Ø SOUND 5ØØØ,1Ø:SOUND 8ØØØ,1Ø:SOUND 5
ØØØ,1Ø
164Ø PRINT@22Ø-.5*LEN(ER$(ER%)),ER$(ER%)
;
165Ø RETURN
166Ø ' *** Get Code Line ***
167Ø PRINT@173,"";:PRINT USING "Scan lin
e ###";NN%
168Ø IF NN%=-1 THEN PRINT@173,"Scan any
line":GOTO 17ØØ
169Ø SOUND 5ØØ,5
17ØØ INPUT#1,IL$:ER%=Ø
171Ø FOR I%=1 TO LEN(IL$)
172Ø IF MID$(IL$,I%,1)="!" THEN MID$(IL$
,I%,1)="."
173Ø NEXT I%
174Ø IF LEN(IL$)<>1 AND LEN(IL$)<>21 THE
N ER%=4:RETURN
175Ø IF LEN(IL$)=1 THEN LL%=Ø:RETURN
176Ø LL$=LEFT$(IL$,2):LL%=(INSTR("Ø12345
6789ABCDEFGHIJKLMNOPQRSTUVWXYZ",LEFT$(LL
$,1))-1)*36+INSTR("Ø123456789ABCDEFGHIJK
LMNOPQRSTUVWXYZ",RIGHT$(LL$,1))-1
177Ø RETURN
178Ø ' *** Open Program File ***
179Ø PN$=LEFT$(IL$,6):IL$=RIGHT$(IL$,LEN
(IL$)-6)
18ØØ OPEN PN$ FOR OUTPUT AS #2
181Ø RETURN
182Ø ' *** Abort ***
183Ø BEEP:BEEP:BEEP
184Ø PRINT@2Ø9,"ABORT!  Are you sure?";
185Ø INPUT#1,AN$
186Ø IF INSTR("YN",AN$)=Ø THEN BEEP:PRIN
T@251,"Scan 'YES' or 'NO'":GOTO 185Ø
187Ø PRINT@2ØØ,SPACE$(8Ø);
188Ø IF AN$="Y" THEN CLOSE:KILL PN$+".DO
":GOTO 149Ø ELSE GOTO 128Ø
189Ø ' *** Skip Line ***
19ØØ IF NN%=1 THEN ER%=6:GOSUB 162Ø:GOTO
 128Ø
191Ø BEEP:BEEP:BEEP
192Ø PRINT@21Ø,"SKIP!  Are you sure?"
193Ø INPUT#1,AN$
194Ø IF INSTR("YN",AN$)=Ø THEN BEEP:PRIN
T@251,"Scan 'YES' or 'NO'":GOTO 193Ø
195Ø PRINT@2ØØ,SPACE$(8Ø);
196Ø IF AN$="Y" THEN NN%=NN%+1
197Ø GOTO 128Ø
198Ø ' *** Stop & Save ***
199Ø BEEP:BEEP:BEEP
2ØØØ PRINT@2Ø7,"STOP & SAVE! Are you sur
e?";
2Ø1Ø INPUT#1,AN$
2Ø2Ø IF INSTR("YN",AN$)=Ø THEN BEEP:PRIN
T@251,"Scan 'YES' or 'NO'":GOTO 2Ø1Ø
2Ø3Ø PRINT@2ØØ,SPACE$(8Ø);
2Ø4Ø IF AN$="Y" THEN 149Ø ELSE GOTO 128Ø
2Ø5Ø ' *** Resume ***
2Ø6Ø IF NN%<>1 THEN ER%=5:GOSUB 162Ø:GOT
```

```
0 128Ø
2Ø7Ø PRINT@254,"Resume Mode";
2Ø8Ø NN%=1:GOSUB 166Ø
2Ø9Ø IF LL%=Ø THEN ER%=5 ELSE IF LL%<>1
THEN ER%=1
21ØØ IF ER%>Ø THEN GOSUB 162Ø:GOTO 2Ø6Ø
211Ø PN$=MID$(IL$,3,6)
212Ø ON ERROR GOTO 214Ø
213Ø OPEN PN$ FOR INPUT AS #2:GOTO 217Ø
214Ø RESUME 215Ø
215Ø CLOSE #2
216Ø ER%=7:GOSUB 162Ø:GOTO 127Ø
217Ø CLOSE #2:OPEN PN$ FOR APPEND AS #2
218Ø NN%=-1:GOTO 128Ø
```

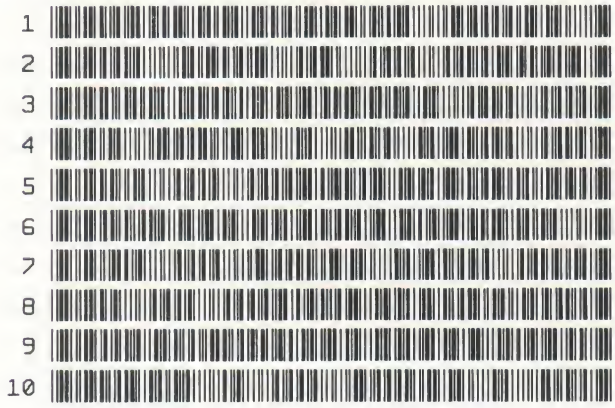
LIST1.BCR (FROM PAGE 56)

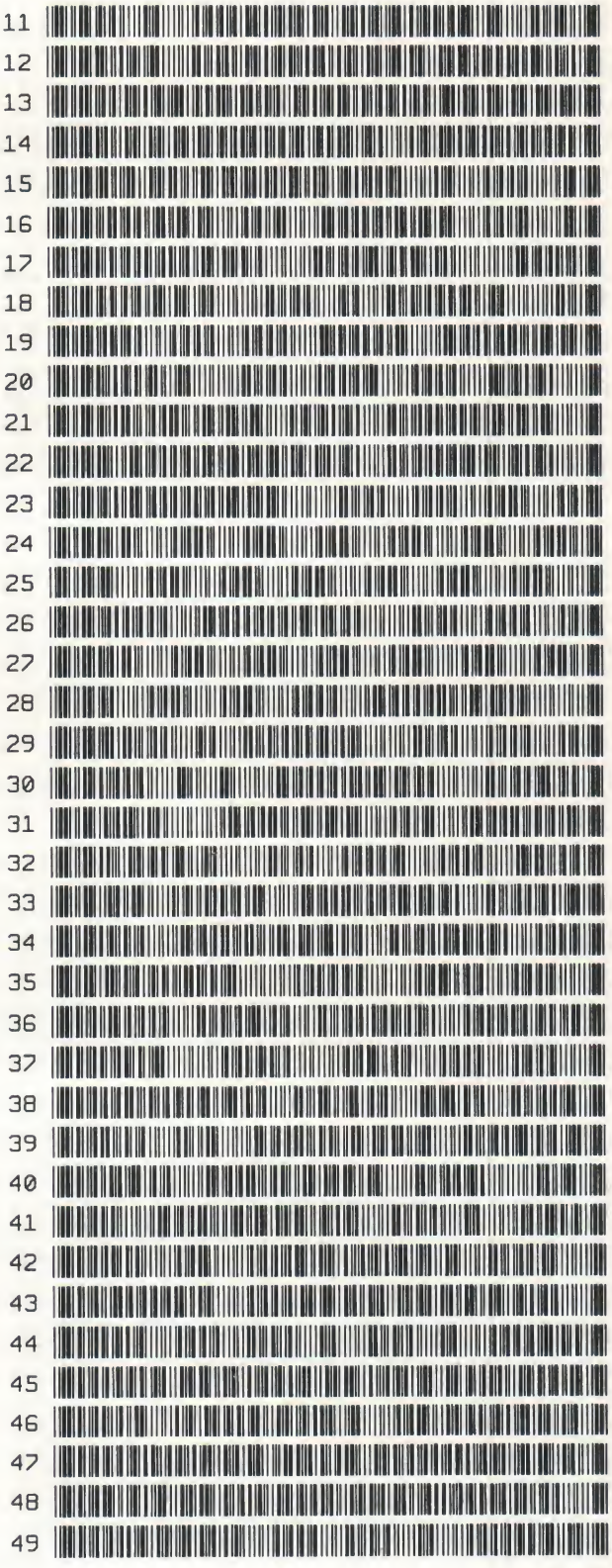


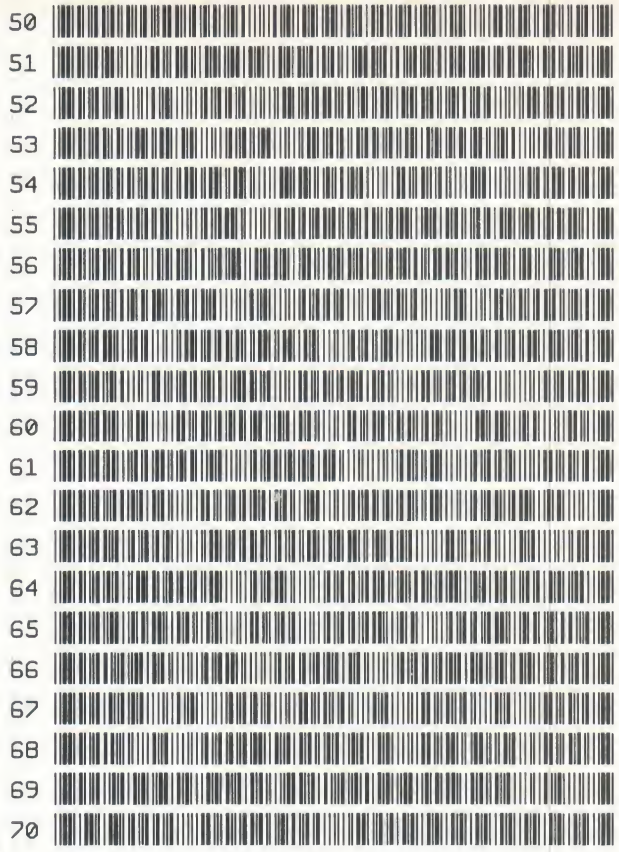
LIST2.BCR (FROM PAGE 56)

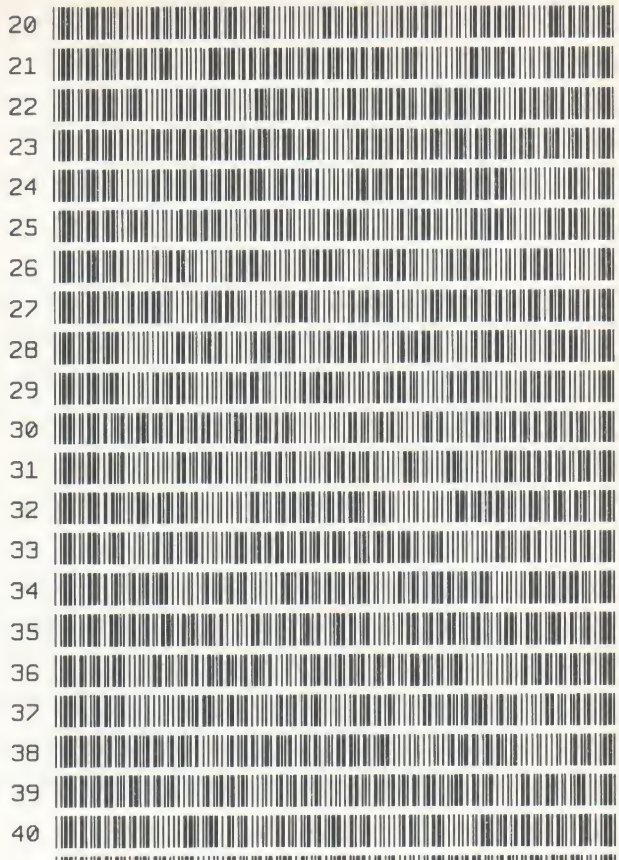


TIMER.BCR (FROM PAGE 31)

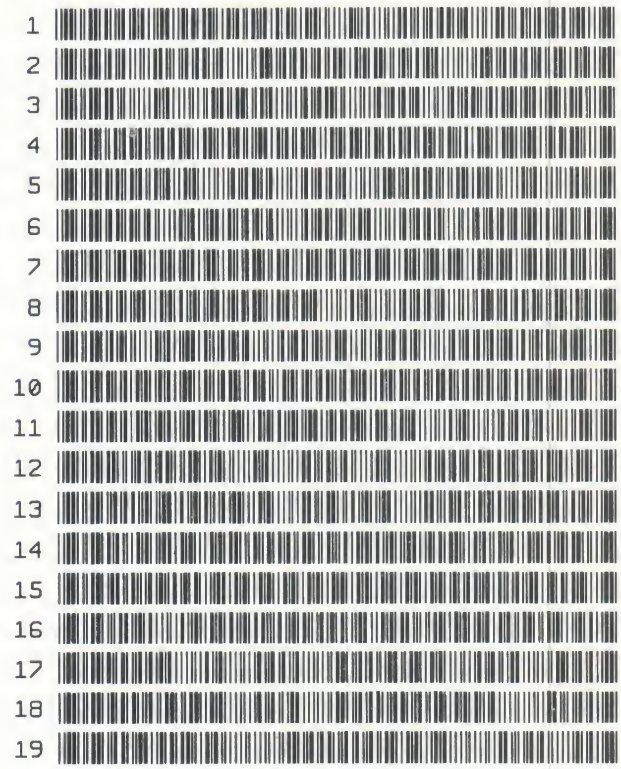


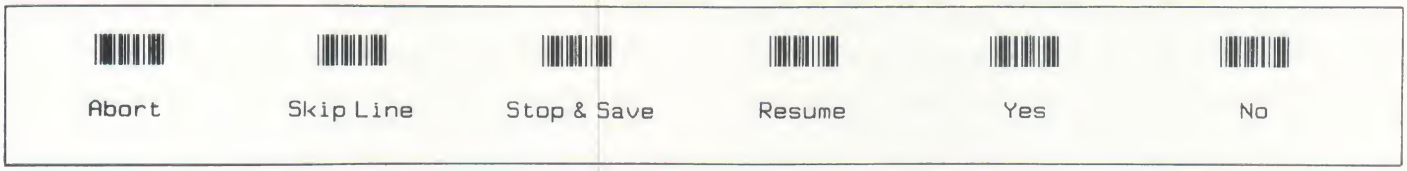


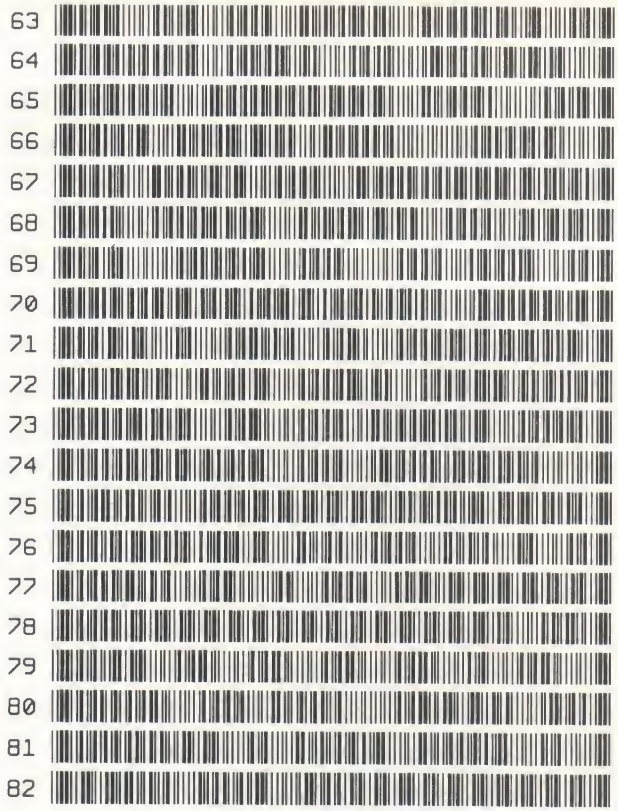




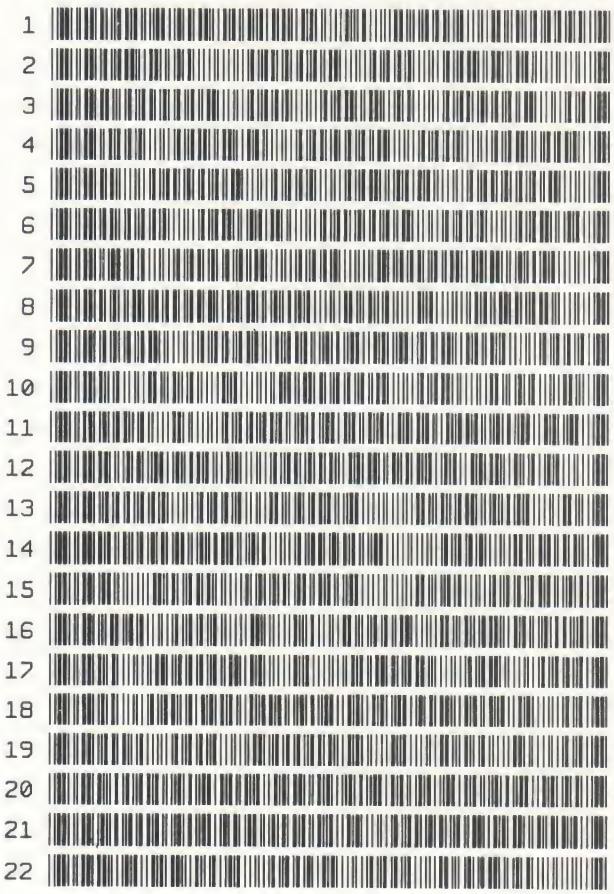
STIMER.BCR (FROM PAGE 31)

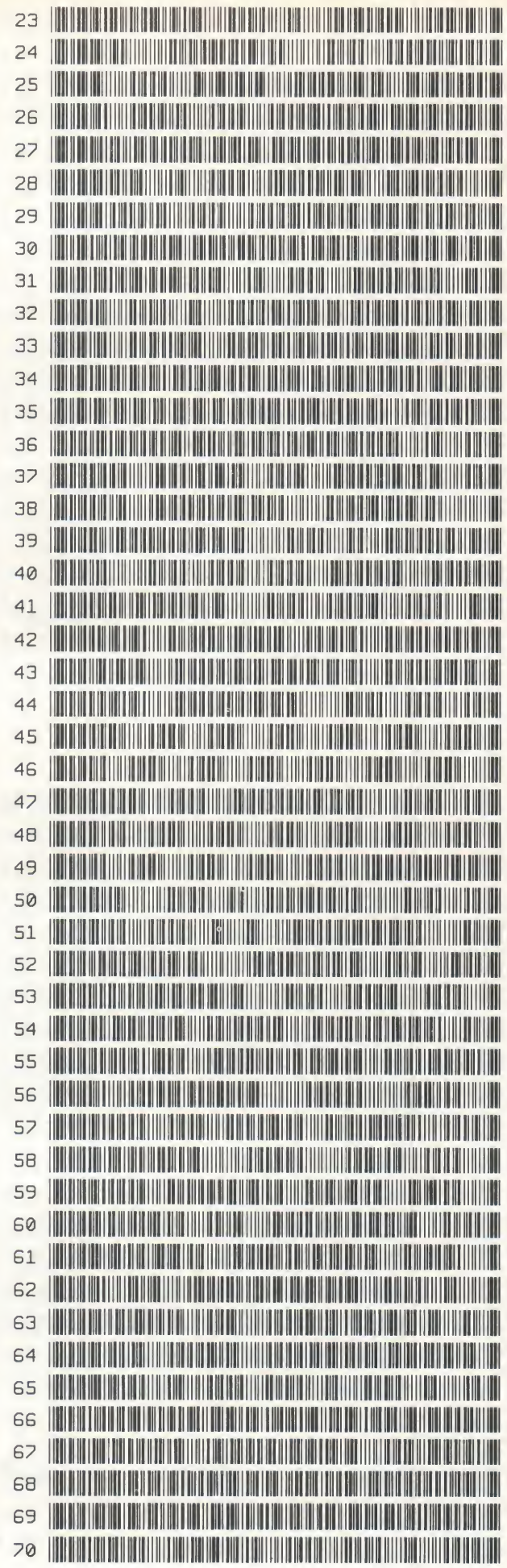


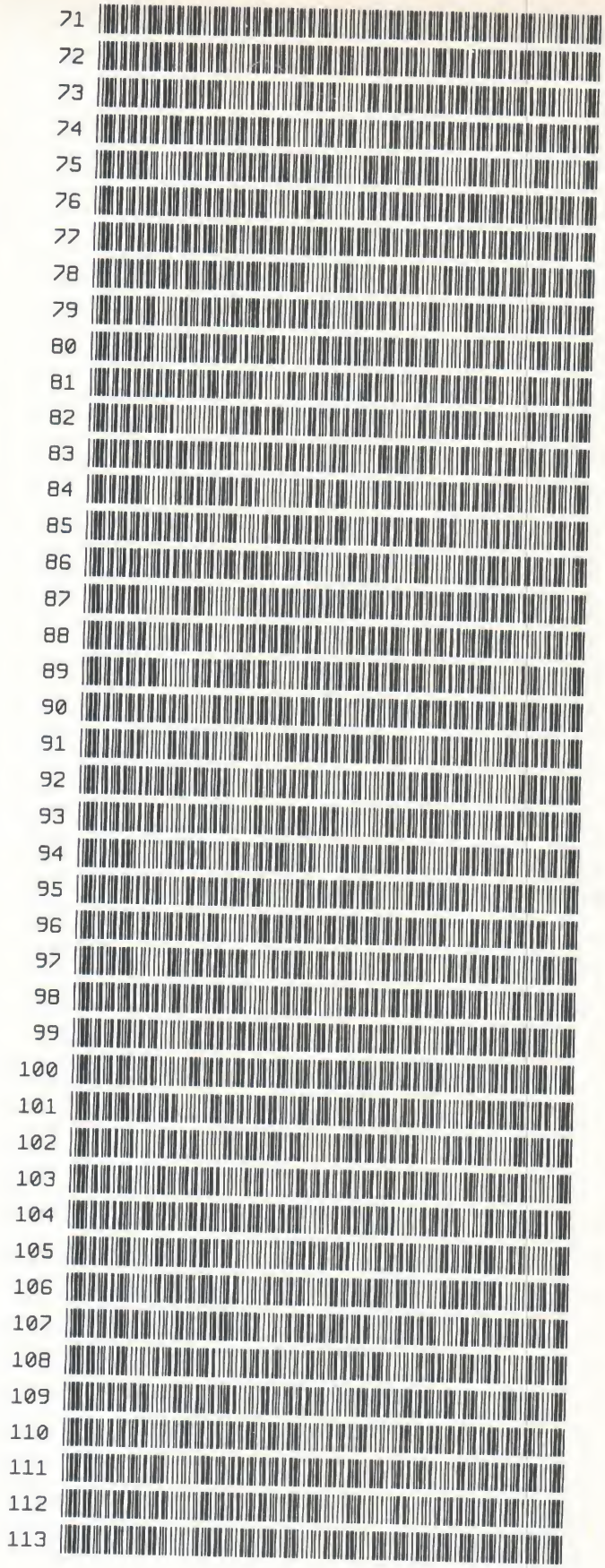


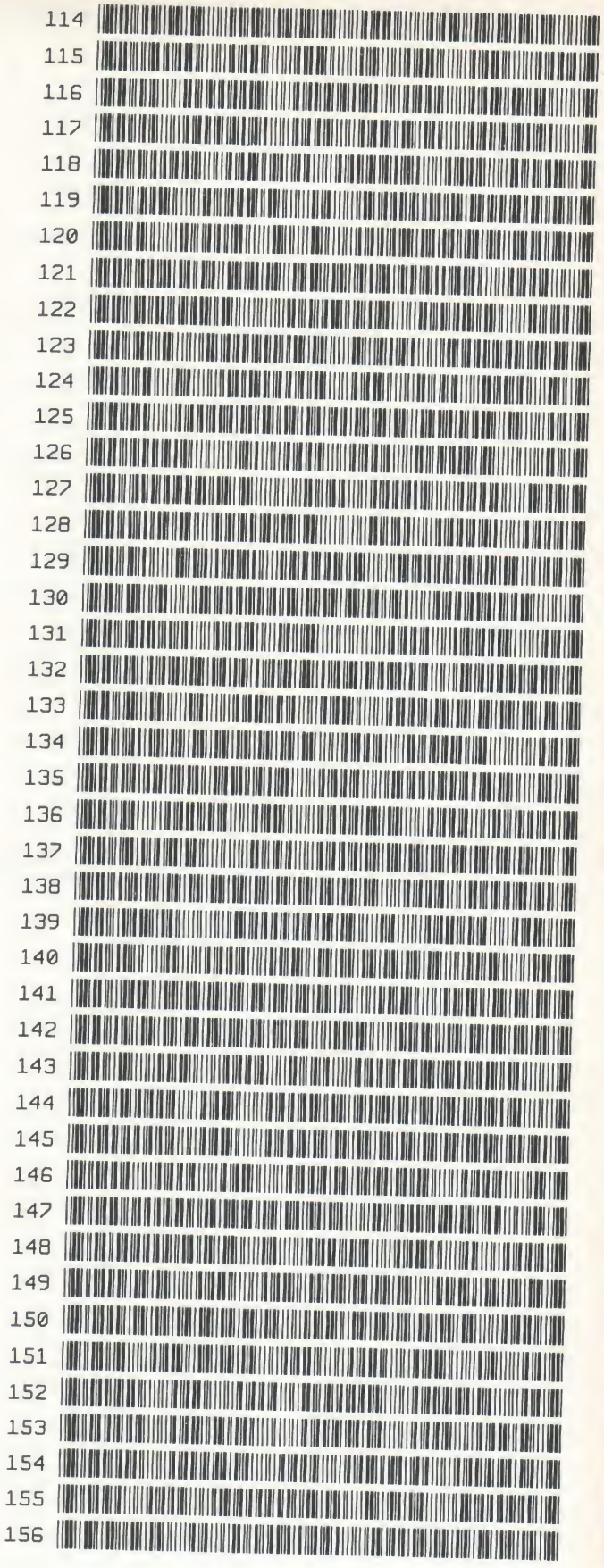


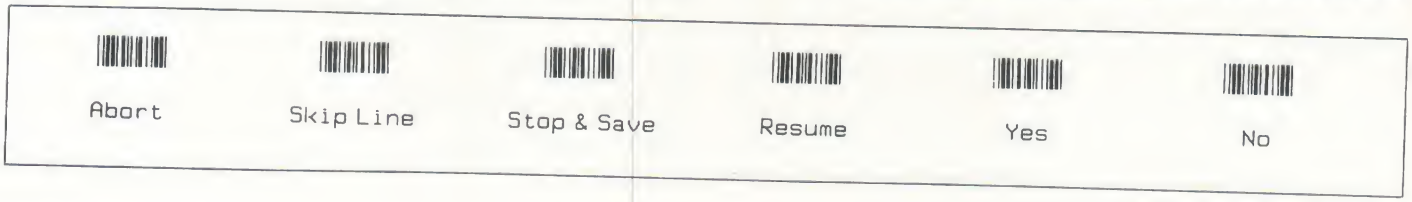
ETIMER.BCR (FROM PAGE 32)






71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113


114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156


Abort
Skip Line
Stop & Save
Resume
Yes
No

The Rackseller

ALABAMA	
Birmingham	Jefferson News Co .
Florence	Anderson News Co.
Greenville	M&B Electronics/Radio Shack
Madison	Madison Books
ARKANSAS	
Fayetville	Vaughn Electronics/Radio Shack
ARIZONA	
Phoenix	Computer Pro
	Tri-Tek
Tucson	Anderson News Co.
CALIFORNIA	
Citrus Heights	Software Plus
Elk Grove	Micro Mainframe
Half Moon Bay	Strawflower Electronics
Livermore	Software Galeria
North Hollywood	Levity Distributors
Santa Rosa	Sawyer's News
Sunnyvale	Computer Literacy
COLORADO	
Salida	Radio Shack
DELAWARE	
Middletown	Delmar Co.
Wilmington	Normar Inc.—The Smoke Shop
FLORIDA	
Boca Raton	Software, Software, Inc.
Davie	Software Plus More
Ft. Lauderdale	Software Connection
N. Miami Beach	Almar Bookstore
Orlando	Book Mania
Panama City	Computer Systems Group
Pensacola	Anderson News Co.
Tallahassee	Anderson News Co.
GEORGIA	
Atlanta	Guild News Agency
ILLINOIS	
Chicago	Chas. Levy Distr.
	Kroch's & Brentano's Jackson Street
	Kroch's & Brentano's Wabash Ave.
	Prairie News Agency
Oakbrook	Kroch's & Brentano's
INDIANA	
Highland	Computer Health Consultants
Scottsburg	Radio Shack of Scottsburg
IOWA	
Ames	Myers TV & Stereo Repair
KANSAS	
Wichita	Lloyd's Radio
KENTUCKY	
Georgetown	Goodwin Electronics
Louisville	Hawley-Cooke Bookstores
Prospect	Falsoft, Inc.
LOUISIANA	
Gretna	Computer Supply Store
Slidell	Radio Shack
MASSACHUSETTS	
Littleton	Computer Plus
Marlboro	Radio Shack
MICHIGAN	
Rochester	Rochester Book Center
Sterling Heights	Software City
	Programs, Inc.
Wyoming	Gerry's Book Co.
MISSOURI	
Independence	Digital Enterprises, Inc.
St. Louis	Software Center
NEBRASKA	
Lincoln	Hobby Town
NEW HAMPSHIRE	
West Lebanon	Verham News Corp.
NEW JERSEY	
Clinton	Micro World II
Hackensack	Total Circulation Service
Pennsville	Dave's Electronic Radio Shack
NEW MEXICO	
Alamogardo	New Horizons Computer Systems
Albuquerque	News and Bookstore
	Page One Newsstand
NEW YORK	
Fredonia	On Line: Computer Access Center
Hudson Falls	G.A. West & Co.
NORTH CAROLINA	
Cary	News Center in Cary Village
Sylva	Total Data Systems
NORTH DAKOTA	
Fargo	Computer Associates
OHIO	
Blanchester	J R Computer Control
Cincinnati	Cinsoft
Kenton	T.W. Hogan & Assoc., Inc.
Mount Orab	Mount Orab Radio Shack
Toledo	Leo's
OKLAHOMA	
Oklahoma City	Merit Computers
OREGON	
Portland	Fifth Avenue News
Woodburn	Pioneer Electronics
PENNSYLVANIA	
Philadelphia	Newsy
Phoenixville	Stevens' Radio Shack
Wind Gap	Micro World
RHODE ISLAND	
Newport	Kelly's Variety
TENNESSEE	
Knoxville	Anderson News Co.
Memphis	Computer Center
Nashville	Mosko's Book Store
Smyrna	Delker Electronics, Inc.
TEXAS	
Bay City	Impulse Computers
Ft. Worth	A & A International
VIRGINIA	
Alexandria	Alonso Book & Period
Norfolk	I-O Computer
Wytheville	Wytheville Electronics
WASHINGTON	
Kennewick	Satellite TV Specialists
Mt. Lake Terrace	Emerald Computer
WEST VIRGINIA	
Lesage	Pioneer Technology
WISCONSIN	
Ladysmith	Electronics, Etc.
Madison	20th Century Books
ARGENTINA	
Cordoba	Informatica Y Telecomunicaciones
CANADA:	
ALBERTA	
Calgary	Rainbow Software Services
Edmonton	CMD Micro
	Kelly Software Distributors
BRITISH COLUMBIA	
Sidney	Sidney Electronics
QUEBEC	
LaSalle	Messageries De Presse Benjamin
NOVA SCOTIA	
Halifax	Atlantic News

Also available at selected B.Dalton Bookseller stores

Advertisers Index

PCM Advertising and Marketing Representatives

Shackleford, Nolan, Davis, Gregg and Associates
Cindy J. Shackleford, president
12110 Meridian South, Suite 5
P.O. Box 73-578
Puyallup, WA 98373-0578
(206) 848-7766

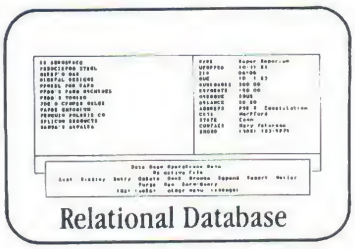
Kim Vincent
Advertising Representative
The Falsoft Building
9509 U.S. Highway 42
P.O. Box 385
Prospect, KY 40059
(502) 228-4492

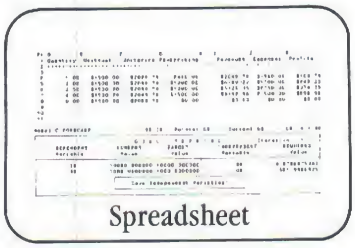
Top-of-the-line: *Open Access*®

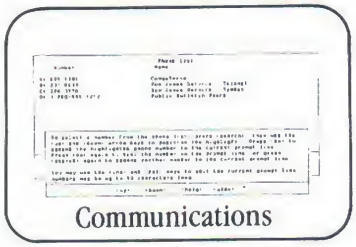
NOW ONLY $395*

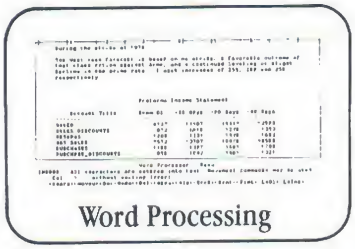
OPEN ACCESS Comparison Chart	Database	Form Query	Report Generator	Query Processor	Relational Database	Graphics	3-D Graphics	Spreadsheet	Goal Seeking	Word Processor	Communications	Time Management	Price
Lotus 1-2-3	✓					✓		✓					$495
Symphony	✓	✓	✓			✓		✓		✓	✓		695
Framework	✓		✓	✓		✓		✓		✓	✓		695
Smart System	✓		✓		✓	✓		✓		✓	✓	✓	895
Open Access	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	**395***

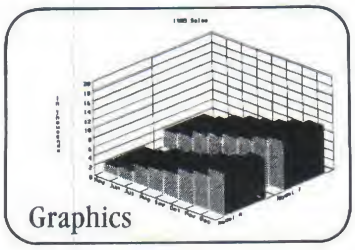
Open Access has sold almost 50,000 copies overseas, and many more in the U.S. thanks to Radio Shack®. With one of the most powerful Relational Databases around, and just about all the additional features you could want, it's no wonder. Just the Goal Seeking feature in the Spreadsheet alone should make *Open Access* your customer's #1 choice.
(We're even written in 11 languages with exact file transfer.)

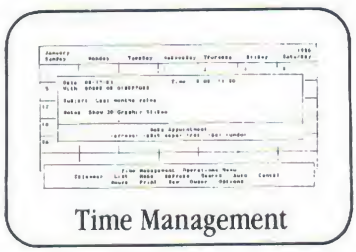

Relational Database


Spreadsheet


Communications


Word Processing


Graphics


Time Management

Available through your exclusive *Express Order Software System*

90-0114 Tandy2000 • 90-0247 Tandy1200 • 90-0437 Tandy1000

The product's exceptional. The price extraordinary. Can you really afford to compromise any longer?

SPI®

SOFTWARE PRODUCTS INTERNATIONAL

10240 Sorrento Valley Road ■ San Diego, CA 92121

*U.S. version only

SUPER ROM

Lucid Spreadsheet | Write ROM | Database | Outliner

"The Tandy Portable Drive with one button boot!"

NOW AVAILABLE ON THE M-200

NOW YOU CAN REALLY HAVE IT ALL!

All on one ROM. Truly the finest four programs available for the Model 100 — guaranteed. Try it for 30 days. If you aren't blown away by the excellence return it for a full refund.

$199.95

The four best programs for the Model 100 all on one ROM. 32K of power without using any RAM for program storage. This is the PCSG Snap-In ROM that just presses easily into the little ROM socket in the compartment on the back. You access the four right from the main menu like built-ins.

Write ROM — the definitive word processor for the Model 100. Function key formatting or dot commands. Search and replace. Library feature — inserts words, phrases or whole documents into text from just a code. MAP lets you see a picture of your document. In all there are 60 features and functions. No one can claim faster operation. FORM lets you create interactive forms with on-screen prompts that you can answer from the keyboard. Nothing else for the Model 100 compares with the features of Write ROM. Exactly the same as the Write ROM sold as a single program. Infoworld says it "makes the Model 100 a viable writing unit . . . surpassed our highest expectations for quality and clarity."

Lucid Spreadsheet: This is the one PICO magazine says "blows Multiplan right out of the socket" and Infoworld performance rated as "excellent" and said "makes the Model 100 compute." Gives you features you cannot get with Lotus 123. Lets you build spreadsheets in your Model 100 that would consume 140-150K on a desktop. Program generating capability with no programming knowledge required. Variable column widths. Includes find and sort with function key control. It's fast, recalculates like lightning. No feature has been taken from the original, only new ones added.

Database: This is a relational data base like no other. You can do everything from mailing lists to invoices. No complicated pseudo-coding, you create input screens as simply as typing into TEXT. You are not limited by size; you can have as large an input screen as you wish. Prints out reports or forms, getting information from as many files as you like. Complete math between fields. Total interface with Lucid worksheets.

Outliner: Does everything that Thinktank does on a PC but a whole lot better. Includes a Sort for your headlines. Lets you have headlines of up to 240 characters. Has cloning, hoisting and sideways scroll up to 250 characters. Like Lucid, this one sets a new standard for outliners. This is the way to plan and organize your projects.

Present Lucid and Write ROM owners can upgrade for $150. If you have both it's $125.

As usual PCSG sells the Super ROM on a thirty day guarantee. If for any reason you are not satisfied, simply return it for a full refund.

We are excited about this product. Super ROM gives the Model 100 the true power of a desktop. No other multi-program ROM has software that compares. But don't take our word for it. We invite you to make that comparison yourself. Priced at $199.95 on Snap-In ROM.

Got stuck with somebody else's multi-ROM? We'll upgrade it for $150.

(214) 351-0564

PORTABLE COMPUTER SUPPORT GROUP

11035 Harry Hines Blvd., #206, Dallas, TX 75229

MC, Visa, American Express, Check, or C.O.D.